SUPER VIC

n° 1 Giugno 1984

L. 3.500

Spedizione in abbonamento postale Gruppo III/70

1

Una pubblicazione della J.soft EDITRICE

SUPERVIC
la rivista per gli utenti dei prodotti Commodore



DISK MENU

MUSIC EDITOR PER C64

LO SCALATORE

ANATOMIA DEL COMPUTER

RE-BEEP

OK-LIST

IDEE E PROGRAMMI PER VIC 20 E C64

PERSONAL COMPUTER: il meglio, in edicola!

BIT
La più letta, la prima e più diffusa.
TEST - 64 Pagine di PROGRAMMI
- Inserto SPECIALE SCUOLA
- E tutte le NOVITA'
HARDWARE E SOFTWARE



PERSONAL SOFTWARE
- L'unica che presenta
software per tutti i personal:
Commodore, Apple, Sinclair, T.I., HP, Sharp, ecc.

Bit e Personal Software sono riviste firmate
GRUPPO EDITORIALE JACKSON

NON UN ABBONAMENTO!

UN SUPERABBONAMENTO

Questo è ciò che la J.*soft* ti offre: un superabbonamento ad una super-rivista!
Una eccezionale offerta di lancio: 16 numeri di SUPERVIC al prezzo di
10 oltre alla possibilità esclusiva di ricevere mensilmente, con la copia della rivista, il supporto magnetico con tutti i programmi pubblicati.
L'offerta è valida fino al 30/6/1984.



LE NOSTRE SUPERPROPOSTE ABBONAMENTI

Abbonamento* a 16 numeri (5 nel 1984 + 11 nel 1985) **di SUPERVIC** al prezzo speciale di L. 35.000 anziché L. ~~56.000~~

Abbonamento* a 16 numeri di SUPERVIC + 16 cassette con tutti i programmi pubblicati al prezzo speciale di L. 75.000, anziché L. ~~120.000~~

***L'abbonamento decorrerà dal prossimo numero**

Abbonarsi è semplice! Effettuate il versamento con l'apposito modulo c.c.p. inserito in questo fascicolo, oppure ritagliate il tagliando abbonamenti pubblicato in questa pagina e speditelo allegando un assegno intestato a:
J.*soft* - Via Rosellini 12 - 20124 Milano.

Tagliando abbonamento a SUPERVIC da inviare in busta chiusa a:
J.*soft* - Via Rosellini 12 - 20124 Milano

☐ Desidero sottoscrivere un abbonamento a 16 numeri (5 nel 1984 + 11 nel 1985) di SUPERVIC al prezzo speciale di L. 35.000 anziché L. ~~56.000~~

☐ Desidero sottoscrivere un abbonamento a 16 numeri (5 nel 1984 + 11 nel 1985) di SUPERVIC + 16 cassette con tutti i programmi pubblicati al prezzo speciale di L. 75.000 anziché L. ~~120.000~~

cognome ____________ nome ____________

via ____________ città ____________

cap. ____________ provincia ____________ data ____________

firma ____________

La piccola guida del principiante

Che cos'è un programma?

Il solo computer non può compiere alcuna operazione.
Un computer possiede potenzialità ma, come una macchina senza benzina, senza un programma, non può funzionare.
La maggior parte dei programmi pubblicati su **SUPERVIC** per i computer Commodore sono scritti in un linguaggio per calcolatori chiamato BASIC. Il BASIC è facile da imparare ed è disponibile, di serie, nel VIC 20 e nel Commodore 64.

Programmi BASIC

Ogni mese, **SUPERVIC** pubblica programmi sia per il VIC che per il 64. Tanto per cominciare, se possiedi un VIC copia solo i programmi scritti per la tua macchina indicati con "versione per VIC 20". Più tardi, quando avrai acquisito esperienza con il BASIC del tuo computer, potrai cercare di copiare e convertire certi programmi per altri calcolatori. Diversamente dal linguaggio corrente, che può essere variamente interpretato, il BASIC di solito ha un solo modo corretto per indicare qualcosa.
Ogni lettera, carattere, o numero ha il suo significato.
Un errore banale è costituito dalla sostituzione nel numero "0" con la lettera "O" oppure il carattere minuscolo "l" invece del numero "1" o ancora il carattere maiuscolo "B" con il numero "8".
Devi anche inserire tutta la punteggiatura, i due punti (:) e le virgole copiando esattamente ciò che appare sulla rivista. Gli spazi possono essere importanti. Per essere sicuro, **copia il listato esattamente** come si presenta.

Le parentesi e i caratteri speciali

L'eccezione per questa regola di copiatura si presenta quando incontrerete indicazioni tra parentesi quali: "[GIÙ]".
Ogni cosa compresa tra parentesi è un carattere speciale oppure un carattere che non può essere facilmente prodotto con la stampante. Incontrando un carattere di questo tipo fate riferimento alla "Piccola guida per l'"input" dei programmi".

Le istruzioni DATA

Alcuni programmi contengono una sezione, o delle sezioni, di istruzioni DATA. Queste linee di istruzione forniscono le necessarie informazioni di cui il programma ha bisogno.
In alcuni casi le istruzioni DATA costituiscono il programma vero e proprio, altri contengono codici grafici. Queste linee sono particolarmente soggette agli errori. Se un solo numero in una linea di istruzione DATA è sbagliato il calcolatore potrebbe "piantarsi" o distruggere il programma. La tastiera e il tasto STOP appaiono inattivi e lo schermo completamente vuoto. Non lasciarti prendere dal panico. Non si è verificato alcun danno.
Per riprendere il controllo devi spegnere il computer e successivamente riaccenderlo. Ciò cancellerà qualsiasi programma presente in memoria per cui è necessario sempre fare il SAVE del vostro programma prima di comandare il RUN.
Se il computer si ferma puoi caricare (LOAD) il programma e cercare l'errore.
A volte, quando il programma viene "lanciato", un'istruzione DATA errata può causare un messaggio di errore.
Il messaggio di errore potrebbe riferirsi alla linea di programma che legge (READ) il contenuto delle istruzioni DATA.

Come conoscere il computer

Dovresti prendere confidenza con il computer prima di procedere alla copiatura del programma.
Impara le istruzioni che si usano per memorizzare e richiamare i programmi da nastro o da disco. Dovrai conservare una copia del tuo programma se non vorrai copiarlo ogni volta che lo devi usare. Impara ad usare le funzioni di "editing" della tua macchina. Come puoi correggere un errore? Puoi sempre ricopiare la linea e in questo caso devi sapere come procedere. Sapresti come inserire i caratteri in "inverse", i caratteri minuscoli e quelli di controllo?
Tutto ciò è spiegato nel manuale del calcolatore.

Un veloce ripasso

1) Copiare il programma una linea alla volta, con ordine. Premere RETURN alla fine di ogni linea. Usare il tasto "DEL" per correggere gli errori
2) Confrontare la linea copiata con quella presente nella rivista. Puoi controllare l'intero programma nel caso in cui si presenti un errore quando esegui il RUN del programma.
3) Accertati di avere inserito le istruzioni tra parentesi graffe con gli appropriati caratteri di controllo (fai riferimento alla **"Piccola guida per l'"input" dei programmi"** che trovi in questa stessa rivista).

Siamo spiacenti di non poter rispondere alle singole richieste di informazioni circa i programmi, prodotti, o i servizi apparsi su SUPERVIC. Nel caso un programma pubblicato contenga un errore tipografico, la correzione apparirà, in una apposita rubrica, su un successivo numero della rivista.

Se avete domande particolari sui programmi pubblicati da SUPERVIC scrivete a:

SUPERVIC
J. soft Editrice
Via Rosellini, 12
20124 MILANO

SUPER VIC

J.soft s.r.l
DIREZIONE, REDAZIONE, AMMINISTRAZIONE
Via Rosellini, 12
20124 MILANO
Tel. (02) 6888228

DIRETTORE RESPONSABILE:
Pietro Dell'Orco

COORDINAMENTO TECNICO:
Riccardo Paolillo

REDAZIONE
Mauro Cristuib Grizzi

HANNO COLLABORATO A QUESTO NUMERO
F. Stella, U. Barzaghi, R. e C. Comini, P. Ferrari

GRAFICA E IMPAGINAZIONE
Luigi Chiesa
Raffaella Toffolatti

FOTOCOMPOSIZIONE:
d&b Via Vignola, 5
Tel. 02/59.85.08
20133 MILANO

CONTABILITÀ:
Giulia Pedrazzini
Flavia Bonaiti

AUTORIZZAZIONE ALLA PUBBLICAZIONE:
Tribunale di Milano n° 201 del 14.04.1984

STAMPA:
Litografia del Sole
Albairate (MI)

PUBBLICITÀ
Concessionario per l'Italia e l'Estero Reina s.r.l.
Via Washington, 50
20046 MILANO
Tel. (02) 4988066 (5 linee R.A.)
Tlx. 316213 REINA I

Concessionario esclusivo per la DIFFUSIONE in Italia e Estero:
SODIP - Via Zuretti, 25
20125 MILANO

Spedizione in abbonamento postale Gruppo III/70
Prezzo della rivista L. 3.500
Numero arretrato L. 6.000
Abbonamento annuo (11 numeri) L. 32.000; per l'estero L. 50.000
i versamenti devono essere indirizzati a:
TechnoClub s.r.l.
Via Rosellini, 12
20124 MILANO
mediante emissione di assegno bancario, cartolina, vaglia o utilizzando il c/c postale n. 19445204
Per i cambi di indirizzo indicare, oltre al nuovo, anche l'indirizzo precedente ed allegare alla comunicazione l'importo di L. 500, anche in francobolli

GLI ARTICOLI TRADOTTI SONO TRATTI DALLA RIVISTA COMPUTE! O DA COMPUTE! GAZETTE COMPUTE! PUBLICATION, INC. P.O. BOX 5406, GREENSBORO, NC 27403 - USA

Per collaborare a SUPERVIC

La maggior parte dei seguenti suggerimenti ha lo scopo di migliorare l'accuratezza e la velocità di pubblicazione di un articolo; rispettando questi consigli si accresceranno le probabilità che un vostro lavoro venga pubblicato. La rivista è interessata ad articoli e programmi riguardanti la linea Commodore. Siamo più interessati al contenuto di un articolo piuttosto che al suo stile, e soprattutto gli articoli devono essere chiari ed esaurienti. La seguente "guida" permetterà che le vostre buone idee e i vostri programmi vengano più facilmente accettati per la pubblicazione;

1 l'angolo superiore sinistro della prima pagina dovrà contenere: nome, cognome, indirizzo, numero telefonico, codice fiscale e data di spedizione.

2 l'angolo superiore destro della prima pagina dovrà contenere la marca e il tipo di computer al quale il lavoro si riferisce, unitamente ad eventuali espansioni di memoria o periferiche richieste.

3 il titolo sottolineato dell'articolo dovrà iniziare a circa due terzi in altezza della prima pagina.

4 le pagine seguenti potranno essere battute normalmente, con la condizione che l'angolo superiore destro contenga un'abbreviazione del titolo e del cognome, unitamente al numero di pagina. Per esempio, Sprite Ed./Brambilla/2.

5 tutte le linee del testo dell'articolo dovranno essere battute con spazio 2 o spazio 3, e un margine di circa un centimetro dovrà trovarsi ad entrambi i lati dello scritto.

6 dovrà essere usata una carta formato A4 e lo scritto dovrà occupare un solo lato del foglio (caratteri maiuscoli e minuscoli).

7 i fogli dovranno essere uniti con una clip.

8 avendo intenzione di spedire più di un articolo, questi dovranno essere inviati separatamente insieme alla rispettiva copia su supporto magnetico.

9 programmi brevi (meno di 20 linee) potranno essere inseriti nel testo, mentre programmi più lunghi dovranno essere listati separatamente. **È ESSENZIALE** per noi disporre di una copia del programma registrata più volte su supporto magnetico, ed anche su entrambi i lati nel caso si usino cassette. È preferibile usare nastri di buona qualità e di lunghezza non eccessiva; la cassetta o il disco dovranno essere etichettati con il nome dell'autore, il titolo dell'articolo, il computer interessato e soprattutto le eventuali espansioni richieste. Come suggerimenti di programmazione, si consiglia di usare le istruzioni "CHR$ (x)", "TAB (x)", "SPC (x)" piuttosto che stringhe di manipolazione del cursore. Ad esempio per uno scroll di 5 linee, l'istruzione "FORI = 1TO5:PRINT:NEXT" è molto più interpretabile di 5 Q inverse; e, invece di una dozzina di simboli di cursore a destra, perché non usare semplicemente "PRINT SPC (12)"? Un rapido controllo dei programmi per operare queste sostituzioni sarà molto apprezzato da noi e dai lettori.

10 per maggior chiarezza, è conveniente usare caratteri maiuscoli riferendosi a istruzioni BASIC (esempio RETURN, LIST, RND, PRINT, etc.). Se si desidera evidenziare una parola, è preferibile sottolinearla piuttosto che scriverla in carattere maiuscolo.

11 gli articoli ed i programmi potranno avere qualsiasi lunghezza - da una routine di una sola linea fino a programmi molto complessi.

12 volendo includere fotografie, queste dovranno essere in formato 24 x 36 o 6 x 6, in bianco e nero o diapositive.

13 non prenderemo in considerazione articoli che siano stati sottoposti ad altre case editrici.

14 il compenso per la collaborazione prestata sarà commisurato alla complessità e all'interesse del programma (da un minimo di L. **50.000** ad un massimo di L. **300.000**). Il pagamento è effettuato in caso di pubblicazione del lavoro.

15 il materiale ricevuto e non pubblicato non verrà restituito.

Spedite i vostri lavori a:

SUPERVIC
Via Rosellini, 12
20124 Milano

e saremo lietissimi di pubblicare i contributi migliori.

La Redazione

EDITORIALE

POINT

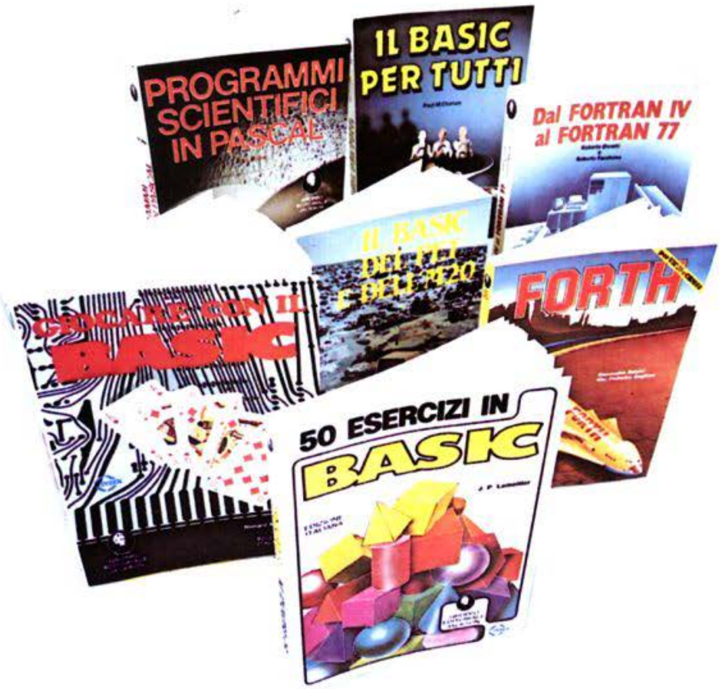
Qualcosa di nuovo!

Eccoci qua! Ci aspettavate? Speriamo di sì!
Gli utenti a cui ci rivolgiamo normalmente sono alla loro prima esperienza in questo campo. Questo determina, oltre ad un grande enstusiasmo, l'impatto immediato con la realtà che si è creata attorno ad un certo tipo di macchine. L'utente che avvicina per la prima volta un personal computer, dopo aver avidamente letto e provato quei piccoli, e volutamente banali, programmi proposti dal manuale, si ritrova a dover cercare, su tutte le riviste di settore, altro materiale da far "girare" sul nuovo amico. La ricerca cui si accennava è talvolta vana e spesso economicamente onerosa.
Ecco dunque **SUPERVIC** nato proprio per risolvere, speriamo in maniera brillante, il problema
SUPERVIC è infatti una rivista dedicata ad un computer specifico e tutti gli argomenti trattati (hardware, software, libri, notizie, etc.) saranno rigorosamente legati a quella macchina o famiglie di macchine.
La rivista, oltre a porsi come naturale complemento all'utilizzo del computer, cercherà di fornire all'utente notizie e informazioni relative alle novità del mercato hardware e software.
I programmi proposti sono selezionati e tradotti dalle migliori riviste inglesi e americane.
SUPERVIC è una rivista indipendente: non legata, cioé, al costruttore o al distributore del computer trattato. Questa scelta garantisce al lettore l'obbiettività necessaria di una rivista che nasce proprio nell'interesse dell'utente.
Inutile dire che auspichiamo la collaborazione dei lettori che, come sempre, sono i giudici assoluti di quanto è proposto; attendiamo dunque giudizi, commenti e quant'altro possa esserci utile per fare la rivista su misura per voi.

Pietro Dell'Orco



Libri firmati JACKSON



La Biblioteca che fa testo

GIOCARE IN BASIC
Il gioco come metodo d'apprendimento del BASIC e dei microcomputer
324 pag. L. **20.000**
Cod. **522A**

PROGRAMMI SCIENTIFICI IN PASCAL
Per costruirsi una "libreria" di programmi in grado di risolvere i più frequenti problemi scientifici e ingegneristici
384 pag. L. **25.000**
Cod. **554P**

DAL FORTRAN IV AL FORTRAN 77
Per chi deve programmare a livello tecnico scientifico e per chi vuole approfondire le conoscenze del linguaggio
266 pag. L. **18.000**
Cod. **517P**

IL BASIC DEL PET E DELL'M20
Un validissimo supporto e strumento di lavoro per chiunque voglia o debba imparare a programmare in BASIC con un Commodore o un Olivetti M20
232 pag. L. **16.000**
Cod. **336D**

FORTH PER VIC 20 E CBM 64
Il libro spiega la programmazione in Forth: linguaggio che dopo essersi affermato in campo scientifico ed industriale, sta ora diffondendosi anche a livello di personal computer.
156 pag. L. **11.000**
Cod. **527B**

IL BASIC PER TUTTI
Per i neofiti una facile e immediata introduzione al linguaggio BASIC e al mondo dei calcolatori
264 pag. L. **17.500**
Cod. **525A**

50 ESERCIZI IN BASIC
Una raccolta completa e progressiva di esercizi matematici, gestionali, operativi, statistici, di svago
208 pag. L. **13.000**
Cod. **521A**

GRUPPO EDITORIALE JACKSON

Attenzione compilare per intero la cedola
ritagliare (o fotocopiare) e spedire in busta chiusa a:
GRUPPO EDITORIALE JACKSON
Divisione Libri
Via Rosellini, 12 - 20124 Milano

CEDOLA DI COMMISSIONE LIBRARIA

VOGLIATE SPEDIRMI

n° copie	codice	Prezzo unitario	Prezzo totale
		Totale	

☐ Pagherò contrassegno al postino il prezzo indicato più L. **2000 per contributo fisso spese di spedizione**

Condizioni di pagamento con esenzione del contributo spese di spedizione:

☐ Allego assegno della Banca ____________ n° ____________

☐ Allego fotocopia del versamento su c/c n. 11666203 a voi intestato

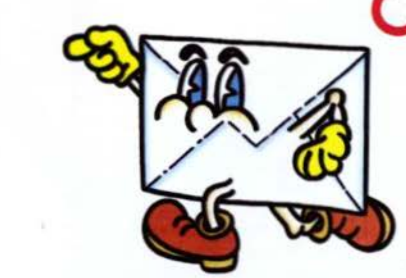
☐ Allego fotocopia di versamento su vaglia postale a voi intestato

Nome
Cognome
Via
Cap Città Prov.
Data Firma

Spazio riservato alle Aziende. Si richiede l'emissione di fattura
Partita I.V.A. | | | | | | | | | | |
ORDINE MINIMO L. 50.000

READ & WRITE

La posta dei lettori



Cari lettori,
come ben potete immaginare è un po' difficile, nel primo numero di una rivista, disporre di lettere che non siano riprese da altri giornali o addirittura inventate di sana pianta: così, solo per questa volta, ci permettiamo di invadere lo spazio che, dai prossimi numeri, sarà occupato da voi. Tutto ciò che abbiamo da dirvi è che... aspettiamo i vostri contributi! E un'altra cosa ancora. Ci piacerebbe moltissimo vedere diventare questa rubrica, più che una botta e risposta tra voi e **SUPERVIC**, uno spazio dove scambiarsi esperienze, trucchi, consigli, fare domande e dare risposte a tutto ciò che può essere di interesse generale.
Siete in difficoltà con il vostro Datassette? Avete scoperto qualche sistema per velocizzare i vostri programmi? La linea x del programma tot provoca la fusione dell'alimentatore? Avete scoperto come stampare un cuore trafitto in reverse? In parole povere, scriveteci per qualunque cosa abbiate da dirci, e pubblicheremo le lettere più interessanti e divertenti. Nell'attesa, vi diamo qualche esempio di "lettere a **SUPERVIC**", naturalmente inventate o copiate con la massima spudoratezza.
A risentirci!

La Redazione

Un bug del C 64

Ho un problema con il mio Commodore 64, che si crea quando uso il tasto DEL per "editare" un progr. BASIC. Mentre uso il DELETE, il programma sul quale lavoro va in esecuzione! Anche se premo il tasto RUN/STOP non si ferma, e quando trova un'istruzione INPUT il computer non accetta nessun carattere. Quando premo un tasto, anche RETURN, non succede niente: il cursore rimane bloccato dove si trova e lampeggia.

R. Kasturi

R. Stai descrivendo l'infausto bug del C 64 circa blocco della tastiera. Accade quando inserisci in basso nello schermo una linea lunga e poi cerchi di editarla. Non c'è modo di uscire dal blocco del computer tranne che spegnendo ed accendendendo la macchina.
Se scrolli in alto la tua linea prima di usare il tasto DEL eviterai ogni problema, in quanto questo bug si verifica solo nella riga più bassa dello schermo.

Dal VIC al 64

Sono possessore di un VIC 20, ma ho deciso di acquistare un C 64; vorrei che rispondeste ad alcune domande a proposito del C 64:
— Tutte le cartridge per il VIC sono compatibili con il C 64?
— Si può usare direttamente il linguaggio macchina sul 64? Ho sentito che si può usare sul VIC 20, anche se è meglio acquistare dell'apposito software su cartridge; il C 64 invece può fare a meno di questa cartridge per programmare facilmente in linguaggio macchina?

J. Smithson

R. Nessuna cartridge per il VIC 20 è compatibile con il C 64, e viceversa. Le porte di ingresso per le cartridge nel VIC e nel C 64 hanno dimensioni diverse, ed anche se potessi inserire la cartridge nella porta, i programmi contenuti non sarebbero comunque compatibili.
Né il VIC né il C 64 richiedono software aggiuntivo per eseguire programmi in linguaggio macchina. Il linguaggio macchina è il linguaggio "di nascita" di tutti i computer — il linguaggio con il quale essi "pensano". Il BASIC, d'altra parte, è una lingua straniera per i computer (come del resto lo è per le persone) e deve essere interpretato interamente al computer prima che questo possa capire le istruzioni. I programmi in linguaggio macchina possono essere caricati ed eseguiti direttamente da cassette, dischi o cartridge sia con il VIC che con il C 64. Possono inoltre essere caricati in memoria da un programma BASIC con istruzioni POKE ed eseguiti con il comando SYS oppure USR.
In ogni caso, per scrivere programmi in linguaggio macchina sul VIC o sul C 64 generalmente occorre del software addizionale. In teoria si può farne a meno, convertendo laboriosamente le istruzioni in codice assembly in numeri decimali e quindi "pokandoli" direttamente in memoria tramite il BASIC. Questo sistema può essere usato per corte routine, ma per programmi più ambiziosi è senz'altro più conveniente l'uso di un MONITOR o di un ASSEMBLER per scrivere in linguaggio macchina.
Alcuni computer (come il Commodore PET e l'Apple) hanno dei monitor interni mentre i computer destinati al mercato degli "home" generalmente ne sono sprovvisti. I monitor e gli assembler si trovano facilmente su cassetta, disco o cartridge sia per il VIC che il C 64.

Le espansioni di memoria del VIC

È possibile aggiungere più di 24Kbytes al VIC 20?

Charles A. Bukey

R. Si e no. Il VIC 20 ha 4 Kbytes di RAM nel suo interno, da 4096 a 8191 ($1000-$1FFF) più 1Kbyte per i puntatori, stack e così via. In totale 5Kbytes RAM. Si possono acquistare espansioni di memoria che aggiungono 3K, 8K, 16K, o 24 Kbyte originali Commodore o di altra produzione.
Se tu programmi in BASIC, 24Kbytes è la massima quantità di memoria che puoi aggiungere al VIC. Più di 35Kbyte, per un massimo di 40Kbytes.
La prima cosa da ricordare quando si aggiunge memoria RAM è che il VIC possiede una sola porta di espansione. Ciò significa che se si possiede un'espansione da 8Kbyte e se ne vogliono aggiungere altri 8Kbyte, si hanno due possibilità. Puoi acquistare altri chip di memoria RAM e ricollegare la cartridge di espnasione (sistema per altro sconsigliato se non si sa esattamente cosa si sta facendo). Oppure si può acquistare uno slot per espansioni multiple che permette di collegare più di una cartridge alla porta delle espansioni. Spesso questi slot sono dotati di interruttori che permettono di selezionare una o più di una delle cartridge collegare. La memoria può essere aggiunta al VIC nei blocchi seguenti:

3Kbytes 1024- 4095 ($0400-0FFF)
8Kbytes 8192-16383 ($2000-3FFF)
8Kbytes 16384-24575 ($4000-5FFF)
8Kbytes 24576-32767 ($6000-7FFF)
8Kbytes 40960-49151 ($A000-BFFF)

READ &WRITE

Aggiungere espansioni di memoria al VIC può causare qualche problema: se si inserisce la 3Kbyte expander, l'area di memoria dello schermo (7680-8191 nel VIC inespanso) rimane allo stesso posto, ma se si aggiungono più di 3Kbyte l'area di memoria dello schermo si muove e 4096-4607. Ciò può creare problemi di compatibilità per programmi scritti per il VIC inespanso. Quando si aggiungono 8Kbyte o più di RAM, tutta la memoria al di sotto dell'area di schermo (4096) diventa invisibile per il BASIC. È per questo che per la programmazione in BASIC si possono aggiungere al massimo 24 Kbyte. Se si aggiunge memoria nelle locazioni 8192-32767, una ulteriore espansione di memoria può essere usata solo per il linguaggio macchina (o PEEK e POKE).

Conversioni del Disk Drive

Possiedo un VIC 20 con un disk drive 1540 e penso di acquistare un C 64. Considerando le differenze tra i drive 1540 e 1541, è emerso che il 1540 può essere cambiato in 1541 sostituendo una ROM. Il mio problema è trovare qualcuno che possa sostituire questo chip al mio drive. Devo rivolgermi direttamente alla Commodore oppure questa operazione può semplicemnte essere fatta a casa?

David Leung.

R. È vero che un drive 1540 può essere convertito in 1541 cambiando un chip ROM. Dopo la conversione il drive funzionerà sia con il VIC che con il C 64. Come regola generale, ogni modifica sull'hardware computer o delle periferiche (come il disk drive) dovrebbe essere eseguita da personale autorizzato. Operare "modifiche casalinghe" può causare danni alla macchina ed invalidare la garanzia.

Sprites, SYS ed immagazzinamento dati sul C 64

Ho parecchie domande da fare sul C 64:

— **Primo**: so che è possibile verificare se uno Sprite è entrato in collisione con un carattere dello sfondo con una PEEK (V+31) dove V=53248. Esiste qualche sistema per sapere quale tipo di carattere è stato interessato alla collisione?

— **Secondo**: ho visto molti programmi con istruzioni SYS (X). Qual'è lo scopo di questa istruzione?

— **Terzo**: dove si può trovare spazio per inserire programmi in linguaggio macchina oltre che in $C000-CFFF (dec. 49152-53247)?

Mattew Price

R. Il byte 53279 di verifica delle collisioni sprite-sfondo è in pratica strutturato come un commutatore: in altre parole registra una condizione ON/OFF (non collisione-collisione). Non offre l'opzione di rivelare con quale carattere è avvenuta la collisione dello Sprite. In ogni caso, tale opzione può essere programmata: ad esempio, dopo che sia stata effettuata una collisione, si può passare ad una subroutine che trasformi le coordinate in pixel dello sprite in coordinate schermo e quindi accedere alla locazione interessata tramite una PEEK che riveli il tipo di carattere.

Il comando SYS è usato per iniziare l'esecuzione di un programma in linguaggio macchina, proprio come il comando GOTO e GOSUB inizia l'esecuzione di una subroutine nel contesto BASIC.

Ci sono molti posti dove inserire in modo sicuro i programmi in linguaggio macchina: $02A7-02FF (dec. 679-767), $033C-03FB (dec. 828-1019) che è il buffer, del registratore, e come già detto, $C000-CFFF (dec. 49152-53247).

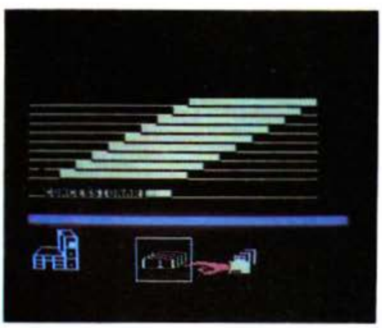


Ovvero come rendere la gestione dei testi facile come un gioco.

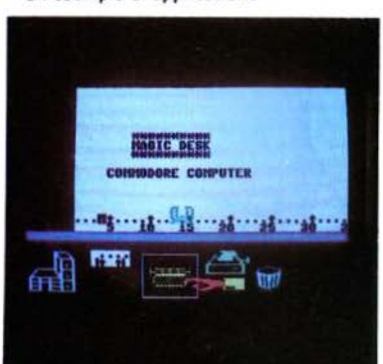
Negli ultimi tempi la tendenza dei costruttori di personal e home computer si è rivolta verso una sempre maggiore semplicità d'uso, per permettere l'impiego produttivo dei sistemi anche da parte di chi non abbia, o non intenda acquisire, alcuna conoscenza di informatica.
Da questa scelta sono nati quindi macchine e programmi "user friendly", cioè "di utilizzo amichevole", in cui tutte le operazioni di gestione dei dati sono automatizzate, mentre le varie possibilità operative sono raffigurate da disegni e sono selezionabili muovendo un cursore sullo schermo a mò di videogioco.
Unico grosso ostacolo alla diffussione di questi sistemi è finora stato il prezzo, solitamente poco abbordabile.
La Commodore ha introdotto in questi giorni sul mercato italiano un programma caratterizzato da una estrema semplicità d'uso e realizzato per il C64. Questo programma si chiama "Magic Desk", e raggiunge l'obiettivo di rendere chiunque non abbia mai visto un calcolatore capace di creare, stampare e archiviare testi nel giro di pochi minuti.
Tutto quello che occorre fare per poter cominciare a lavorare è inserire la cartuccia su cui risiede il programma e accendere il calcolatore.
Immediatamente appare sullo schermo il disegno a colori di un ufficio, con una scrivania, una macchina da scrivere, uno schedario e un cestino della carta straccia.
Nella parte superiore del disegno è raffigurato anche un dito indice, che si muove comandato da un joystick.
Per creare un testo basta toccare con il "dito" la macchina da scrivere e premere il pulsante del joystick.
A questo punto la scena cambia e lo schermo ci presenta una macchina da scrivere con un foglio bianco già inserito.



La schermata principale



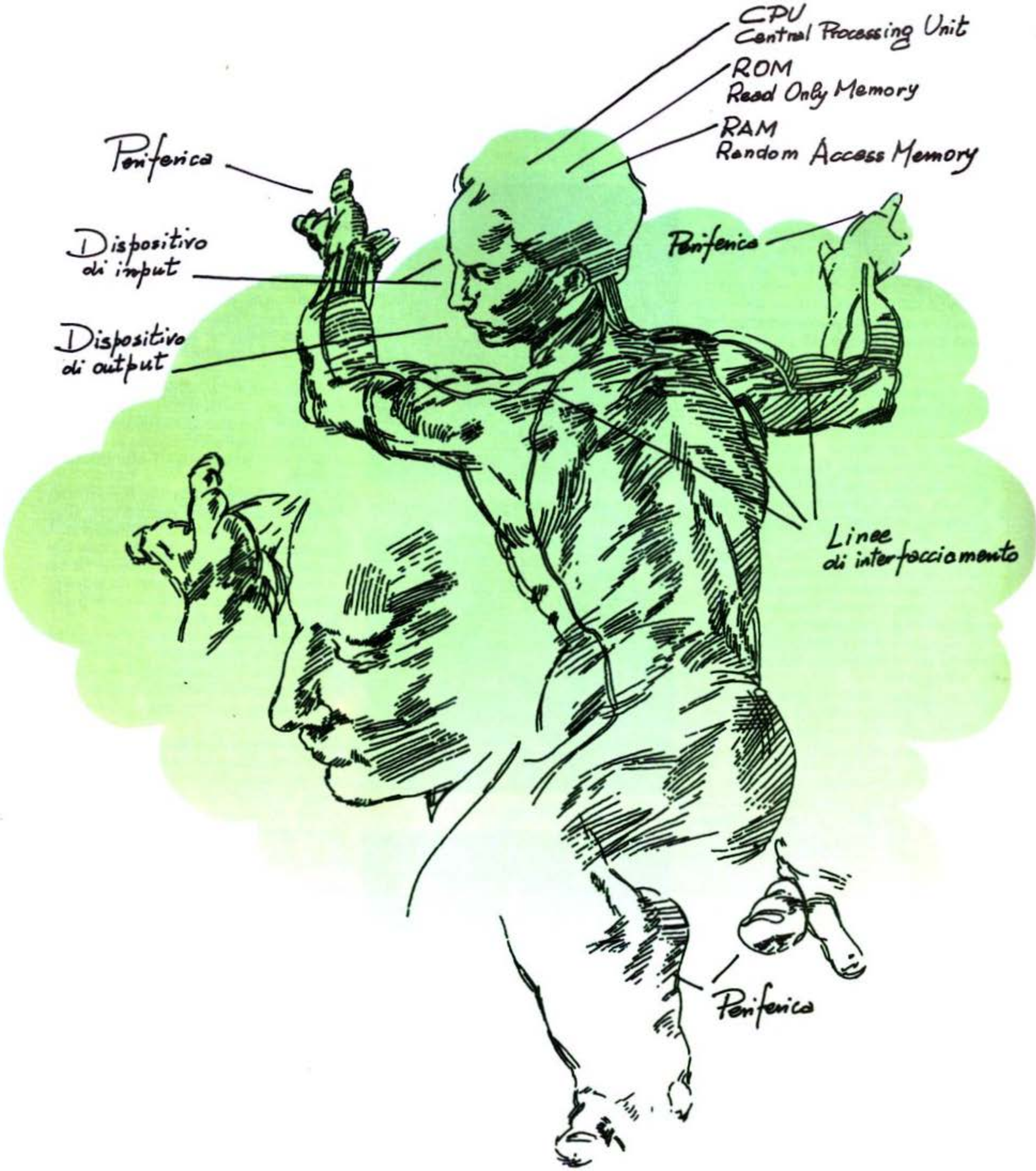
Un esempio di applicazione



La "macchina da scrivere" Magic Desk

Questa macchina da scrivere "simulata" possiede tutte le funzioni di quelle vere (marginazione, tabulazione, eccetera), selezionabili sempre per mezzo joystick.
Per stampare il testo creato é sufficiente indicare la stampante, mentre per archiviarlo su dischetto bisogna "prendere" una cartelletta dallo schedario, scrivere sull'"etichetta" il nome del contenuto e "riporre" il tutto.
Per cancellare un testo è sufficiente "gettarlo nel cestino", dove cade con un tonfo.
Tutto questo è possibile perché Magic Desk si occupa di tutti gli aspetti "informatici" del lavoro, come formattare i dischetti, aprire e chiudere i files, gestire i vari parametri di input/output eccetera.
Anche i messaggi di errore sono molto chiari, e contengono una serie di suggerimenti per risolvere la situazione.
Quando serve aiuto, in qualunque circostanza basta premere il tasto "Commodore" per ottenere una lista dettagliata di istruzioni relative alla parte del lavoro che si sta svolgendo.
È particolarmente difficile con questo programma distruggere per errore il lavoro fatto in quanto ogni comando che implichi la cancellazione di dati deve essere confermato almeno due volte.
Attualmente il Magic Desk è in grado di creare, stampare e archiviare testi, ma saranno presto in distribuzione versioni in grado di gestire congiuntamente anche indirizzari, rubriche telefoniche e contabilità.
Oltre al prezzo contenuto e alla capacità di render una persona operativa sul calcolatore in tempi brevissimi il Magic Desk possiede anche il pregio di smitizzare e di rendere simile a un gioco il primo approccio con l'informatica, mettendo il neoutente in grado di utilizzare in futuro programmi più complessi e articolati.

MAGIC DESK
è importato e distribuito da:
COMMODORE ITALIA S.p.A.
Via Gracchi, 48
20092 Cinisello Balsamo (Milano)


CPU
Central Processing Unit
ROM
Read Only Memory
RAM
Random Access Memory
Periferica
Dispositivo
di input
Dispositivo
di output
Periferica
Linee
di interfacciamento
Periferica

di Tom R. Halfhill
trad. e adatt. F. Stella

Anatomia di un computer

È più facile capire i computer quando si conosce cosa scandisce il ritmo al loro incessante lavorio. Ecco come le varie parti di un computer lavorano, con invisibile ma continua sintonia, per creare i risultati che poi tutti vedono sullo schermo.

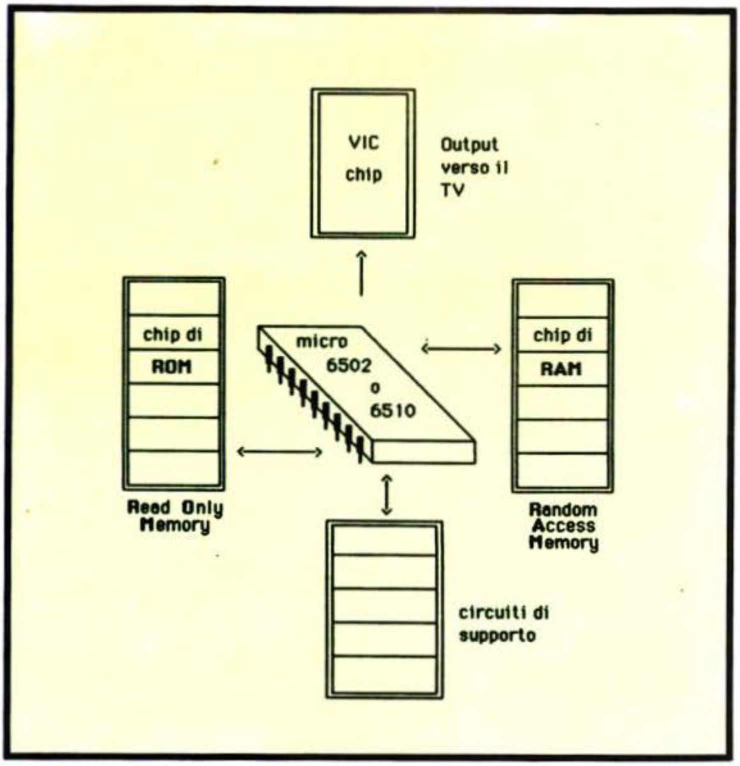
Non è necessario essere ingegnere o meccanico per guidare un'automobile, così (come) per usare un personal computer non è richiesta nè la laurea in elettronica nè la qualifica ufficiale di programmatore. Ma una sommaria conoscenza di ciò che accade sotto il coperchio del vostro VIC 20 o Commodore 64 può essere molto utile quando si cerca di scoprire come funziona, o perché non funziona, un certo programma. Un messaggio banale come **"?DEVICE NOT PRESENT ERROR"** ha un senso molto più chiaro quando si comprende come il computer interagisce con i vari dispositivi periferici. E, persino a chi non è interessato alla programmazione, una comprensione, seppure elementare, del funzionamento di un computer può chiarire perché certi comandi siano indispensabili per utilizzare correttamente i programmi, evitando così l'impressione che le istruzioni siano semplicemente il frutto dell'immaginazione di chi ha scritto il manuale.

Per molti versi il computer è simile al cervello umano, almeno sotto il profilo strutturale; come il cervello, che noi definiamo un organo, sebbene sia un insieme strettamente interconnesso di strutture indipendenti fra loro, il computer è un sistema aggregato, composto da vari sottosistemi. A differenza del cervello, però, che può eseguire più operazioni contemporaneamente, la maggior parte dei computer ne può eseguire una alla volta. Questo può sembrare incredibile, soprattutto dopo aver provato alcuni giochi di azione nei quali decine di oggetti delle più svariate forme e colori sembrano sfrecciare sullo schermo simultaneamente, ma è vero. I computer danno solo a vedere di svolgere più compiti contemporaneamente grazie alla loro velocità di esecuzione. Anche il più lento home computer può eseguire centinaia di migliaia di istruzioni al secondo. La simultaneità è quindi solo un'illusione.

Il "cervello" principale di un computer è il processore **CPU** (da Central Processing Unit); la CPU esegue o coordina tutte le sue funzioni principali. Il compito fondamentale della CPU è raccogliere una per una le istruzioni del programmatore, calcolare le espressioni numeriche, immagazzinare i risultati e passare all'istruzione successiva.

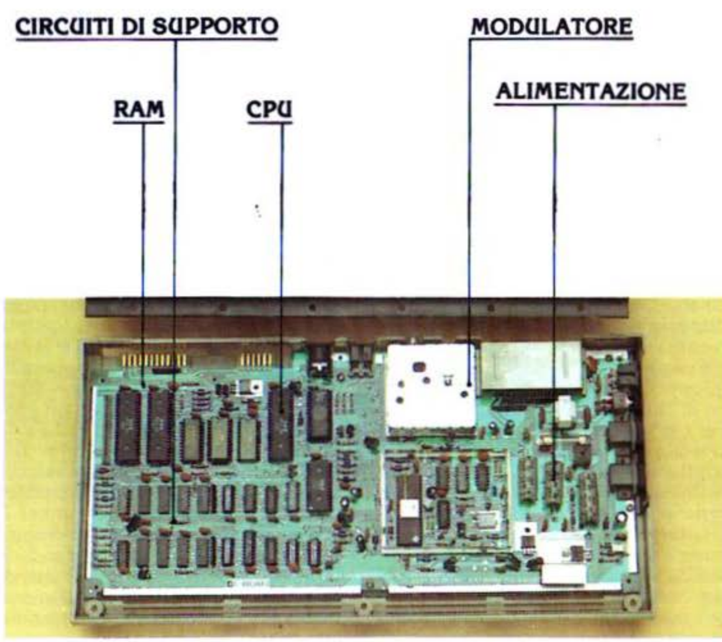
Nel passato la dimensione della CPU era tale da condizionare in modo determinante la dimensione del computer; ancora oggi la CPU di un computer molto grande può occupare una scatola con dimensioni pari a quelle di un frigorifero. Nei primi esemplari, costruiti negli anni cinquanta e sessanta, la CPU occupava una intera stanza od anche un intero fabbricato. Il balzo tecnologico che ha reso possibile la costruzione e la diffusione dei personal computer si é concretizzato nel 1971 quando i progettisti riuscirono a concentrare una intera CPU su di una sola scheggia di silicio più piccola di un'unghia. Da questo "microprocessore" nacque il "microcomputer". Generalizzando, si definisce microcomputer il calcolatore che, usando un microprocessore come CPU, si sistema con facilità su di una scrivania e costa meno di 25 milioni. Ci sono molti tipi di microprocessori; quello del VIC 20 è identificato con la sigla 6502 e fu disegnato al principio del 1970 dalla MOS Technology, ora consociata della Commodore. Il 6502 è abbastanza diffuso grazie alla sua elevata velocità ed al suo basso costo (meno di 5$ acquistato in quantità). I Commodore PET, superPET e la serie 8000 così come l'Apple e l'Atari, usano la CPU 6502. Il Commodore 64 usa una versione più recente del 6502 chiamata 6510, identica nei particolari più importanti a quella precedente.

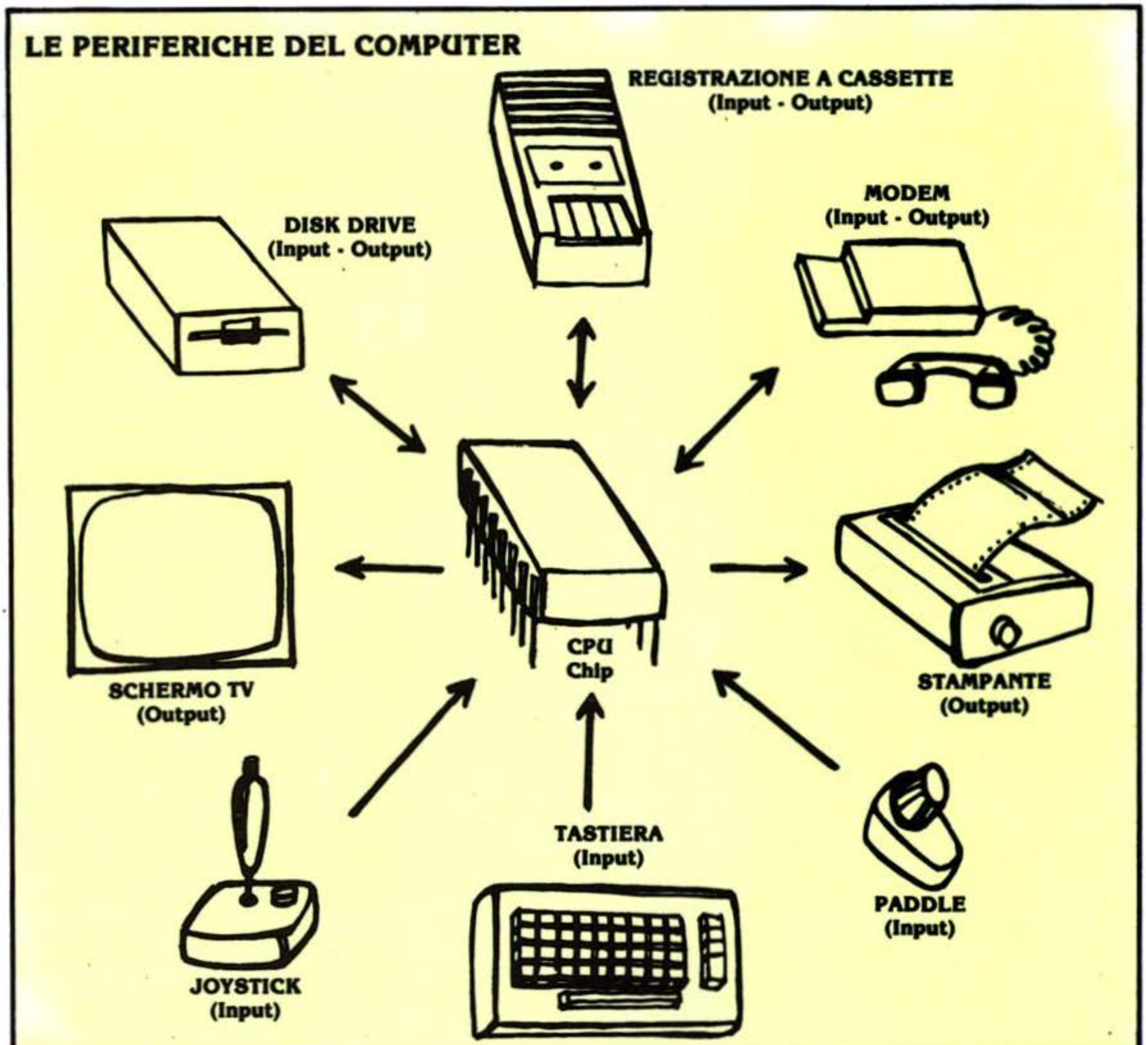
Ma la CPU da sola non fa un computer: da sola è come un cervello senza corpo, un piccolo nucleo di "intelligenza" senza scopo; perciò la CPU deve essere collegata ad un certo numero di altri sottosistemi prima di poter essere qualificata come COMPUTER. Quello più importante è la memoria; la CPU già ne comprende una piccola quantità, che utilizza per propri scopi e non sufficiente per le normali applicazioni di un sistema. Così la CPU è collegata ad un gruppo di chips di memoria. Le memorie hanno subito la stessa evoluzione delle CPU, partendo dall'ammasso di valvole, che riempivano stanze intere, ai transistor, per arrivare agli attuali circuiti integrati. Nei personal computers ci sono in generale due tipi di memoria: Random Access Memory (memoria ad accesso casuale abbreviata in **RAM**) e Read Only Memory (memoria a sola lettura, **ROM**). C'è una differenza fondamentale tra memoria RAM e memoria ROM, la prima necessita di un flusso costante di elettricità per mantenere le informazioni, mentre la seconda è permanente e mantiene il suo contenuto anche quando non è alimentata. Ogni volta che il computer è spento o subisce un calo di tensione, anche per una frazione di secondo, tutto ciò che era stato immagazzinato nella memoria RAM si cancella. Riaccendere il computer non serve a ripristinare le informazioni perdute. Per di più la memoria RAM è quella dove vengono immagazzinati i programmi per la successiva esecuzione. Per questo motivo, ogni volta che si accende il computer è necessario caricare i programmi desiderati da nastro, da floppy disk oppure inserendo una cartridge. Il programma, una serie di istruzioni per la CPU, è immagazzinata permanentemente su uno di questi supporti; il comando LOAD copia il programma nella me-



Se si è poco abituati a ragionare in termini di numerazione binaria è meglio pensare al byte come ad una unità che può rappresentare un carattere, un simbolo, una lettera. Un chilobyte di memoria può contenere circa mille caratteri. Comparare i Kbyte di memoria RAM è un modo di apprezzare la potenza dei vari computer, ma non il solo. Il VIC 20 è fornito con 5 Kbyte di memoria RAM, espandibile a 32 Kbyte, ed il Commodore 64 ha 64 Kbyte di RAM. Questo non significa, però, che il VIC senza espansione sia 12 volte meno potente del C64; è opportuno ricordare che molte altre caratteristiche devono essere prese in considerazione, e che alcune di queste potrebbero essere più importanti per alcuni che per altri.
Dotato di una CPU e della memoria un computer è all'incirca equivalente al cervello umano, ma per rendere un computer (od un cervello) realmente utilizzabili sono necessari molti altri componenti. Come il corpo è dotato di appendici il computer è dotato di periferiche; queste sono costituite da dispositivi collegati alla CPU che le permettono di comunicare con l'ambiente esterno e soprattutto con gli esseri umani. Le periferiche possono essere di due tipi: di **INPUT** (ingresso) o di **OUTPUT** (uscita), anche se alcune riuniscono entrambe le possibilità. Molti utenti sono così abituati ad alcuni di questi dispositivi che arrivano a non pensarli come reali periferiche; ad

moria RAM. La ROM, invece, è completamente differente: le informazioni sono permanentemente impresse nei "chips" dal costruttore cosicché non è possibile cambiare o cancellare in alcun modo il loro contenuto. Le ROM sono quindi usate per conservare quelle informazioni che il computer utilizza sempre e che non devono essere "dimenticate". Memorie di tipo ROM sono anche quelle contenute nelle cartridge poiché alcuni costruttori preferiscono registrare programmi di uso frequente, come i giochi, sulle cartridge, invece che su nastro o disco. Inserendo la cartridge nell'apposita presa, le ROM, in essa contenute, divengono parte del computer cosicché non vi sono tempi d'attesa per il caricamento in memoria del programma.
La quantità di memoria disponibile in un computer dipende dal numero e dal tipo di chips impiegati; quando si comparano fra loro computers sulla base della memoria normalmente si parla di RAM, perché maggiore è la memoria RAM a disposizione e più grande può essere il programma e/o la quantità di informazioni trattabili, cosa che rende il computer più potente. Per consentire una facile comparazione, la memoria è misurata in chilobyte (Kilobyte abbreviato Kbyte); un chilobyte equivale a 1024 byte, un byte, a sua volta, è composto da otto bit, o cifre binarie.





esempio la tastiera è soggetta a questo "rischio". La CPU esplora questa importante periferica con cadenza molto serrata per verificare se qualche tasto sia stato premuto; quando ciò succede, un codice numerico è immagazzinato in una certa area di memoria che la CPU legge comportandosi di conseguenza; la prima operazione che esegue è il mostrare il carattere prescelto su uno dei suoi dispositivi di OUTPUT, il televisore. La tastiera ed il televisore sono le due basilari periferiche utilizzate. Uno dei tasti più importanti è il RETURN, il cui nome deriva dal "carriage return" della macchina da scrivere; oltre a portare il cursore a capo la funzione svolta dal tasto RETURN è anche, e soprattutto, di comunicare al microprocessore che la riga appena terminata è da interpretare ed eseguire. Ad esempio, digitare il comando LOAD, senza premere il tasto RETURN, causa l'apparizione sullo schermo delle lettere L-O-A-D, ma la CPU le ignora e resta in attesa; premendo poi il RETURN appare immediatamente il messaggio **"PRESS PLAY ON TAPE"**. Ciò dimostra che la pressione del tasto RETURN qualifica le lettere digitate e apparse sullo schermo come comando, nel caso specifico utilizzato per caricare programmi da cassetta. La CPU, allora, verifica i tasti del registratore come fa per la tastiera, scopre che il tasto PLAY non è premuto e visualizza la richiesta appropriata; prima del RETURN il computer restava in attesa permettendo così all'operatore di cancellare, correggere o cambiare idea in qualsiasi momento.

Il registratore a cassetta è un'altra periferica molto familiare e funziona come dispositivo di INPUT, ad esempio nella fase di LOAD, e di OUTPUT nella fase di SAVE. Ci possono essere altre periferiche collegate a costituire il "sistema"; la stampante è, ovviamente, un dispositivo di OUTPUT mentre il joystick e le paddle sono dedicati all'INPUT. Il "floppy disk drive" permette di sostituire il nastro magnetico con un piccolo disco flessibile ed ottenere una maggiore velocità di lettura e scrittura; il MODEM (MOdulatore-DEMOdulatore) permette di collegarsi alla rete telefonica e colloquiare con altri computer. È facile capire che, anche la più potente CPU per quanto dotata di ampia disponibilità di memoria sarebbe impotente senza periferiche; il computer sarebbe cieco, sordo e muto.

Fino ad ora abbiamo scritto solo di hardware, cioé delle parti che è possibile vedere e toccare; ma l'hardware è

Anatomia di un computer

un insieme di chips di silicio circuiti stampati, fili, metallo e plastica; è come un corpo con un cervello senza vita, manca ancora qualcosa per farlo lavorare: il **SOFTWARE.**

Il software è l'insieme dei programmi che dicono alla CPU cosa fare e come farlo. Il software è per il computer quello che per il cervello umano è l'intelletto. Se il software è scadente, il computer può sembrare stupido, ma il problema è in realtà il mancato sfruttamento delle potenzialità disponibili; è come un uomo intelligente ma privo di mezzi culturali. Non importa quindi quanto è valido il computer: non apparirà mai migliore di quanto possa dimostrare il suo miglior software. Ecco perché la scelta del software è un argomento molto delicato. Infatti, se si ha in mente una particolare applicazione quando si decide di acquistare un computer è forse più razionale cercare prima il software adeguato e poi scegliere la macchina che lo supporta, quale essa sia. Sfortunatamente alcuni seguono il criterio opposto e finiscono con l'essere delusi. Ci sono due categorie di software, ed una di esse è praticamente invisibile per la maggior parte degli utilizzatori; i professionisti dell'informatica, quelli che lavorano con i "maxicomputer", si riferiscono a questi due tipo definendoli software applicativo: giochi, programmi educativi, word processor, che per quanto diversi fra loro sono accomunati dall'unico scopo di permettere di "fare" qualcosa con il computer. D'altra parte sarà invece difficile trovare qualcuno che abbia familiarità con il software di base, la maggioranza probabilmente non ne conoscerà nemmeno l'esistenza perché questo software agisce "dietro le quinte" eseguendo operazioni che non attraggono l'attenzione dell'utente; eppure, se non ci fosse, il calcolatore non potrebbe funzionare. Tornando alla nostra analogia con il cervello umano possiamo paragonare il software di sistema al sistema nervoso simpatico, involontario, in ogni momento della vita il cervello esegue i compiti di routine che ci mantengono in vita: regola il battito cardiaco e la respirazione, coordina il funzionamento di ogni organo dalla ghiandola pituitaria ai reni e mantiene anche in ordine se stesso disponendo l'utilizzo di celle per la memoria ed altri scopi. Eppure, tutte queste sofisticate operazioni, che richiedono stanze piene delle nostre migliori macchine per essere duplicate artificialmente, vengono eseguite senza il minimo pensiero cosciente. Se avessimo da comandare conscientemente tutto il funzionamento del nostro corpo non potremmo fare nient'altro. Regolando sullo sfondo delle nostre normali attività queste funzioni, il cervello ci permette di fare o pensare quello che preferiamo.

Il computer lavora esattamente nello stesso modo. Il software di base, ed in particolare il sistema operativo, svolge tutti i compiti piccoli ma significativi che concorrono al funzionamento del computer; senza doverci nemmeno pensare ci vediamo risparmiato un grosso e noioso lavoro che il sistema operativo svolge automaticamente anche quando altri programmi sono in macchina. Per esempio, salvando o caricando un programma il S.O. (Sistema Operativo) si prende cura tutti i numerosi dettagli coinvolti nella trasmissione di dati alle periferiche. Digitare i comandi SAVE e LOAD può sembrare elementare, ma solo perché il sistema operativo sta facendo il grosso del lavoro in modo discreto e poco appariscente. Tutto ciò è paragonabile al collegamento tra cervello e braccia, è possibile muovere gli arti con un pensiero cosciente ma è inconscio il coordinamento delle decine di muscoli e tendini coinvolti nell'operazione; a causa del fatto che non ci rendiamo conto in dettaglio di ciò che accade, lo diamo per scontato; e così lavora il sistema operativo; esattamente come il cervello è il S.O. che mantiene in ordine la memoria e impedisce che le informazioni si cancellino l'una con l'altra sovrapponendosi. In questo caso il sistema operativo lavora in stretta connessione col **BASIC**, è un programma come gli altri che lavora in modo indivisibile come fa anche il **DOS** (Disk Operating System) che, in stretta connessione con il computer, gestisce l'unità disco nelle operazioni di INPUT/OUTPUT.

Il software di base ovviamente non è scritto in BASIC, che anzi è frutto di questa programmazione, è scritto invece in linguaggio macchina, un linguaggio direttamente comprensibile dal microprocessore che consiste di un insieme di comandi codificati sotto forma binaria. Ogni tipo di CPU ha il suo linguaggio costituito da un set di istruzioni, normalmente incompatibile con le altre, e non comprende il BASIC come non comprende la lingua inglese o quella italiana. Quando si lancia un programma scritto in BASIC, il BASIC traduce o più propriamente **INTERPRETA** ogni comando per la CPU convertendolo in linguaggio macchina. Normalmente tutto ciò è talmente rapido che non è possibile rendersi conto di quello che accade. In ogni caso programmi molto complessi, come giochi d'azione con continui effetti di movimento, sono molto lenti se scritti in BASIC; questo è il motivo per cui la maggior parte dei programmi commerciali è scritta direttamente in linguaggio macchina, che è molto più difficile della programmazione BASIC ma, evitando l'interpretazione di ogni comando, rende l'esecuzione decisamente più veloce. Se il SISTEMA OPERATIVO, il BASIC e il DOS sono programmi, sarebbe giusto chiedersi come mai non ci si ricorda di averli mai caricati in memoria. La risposta sta nella differenza già illustrata tra RAM e ROM, perché il software di base é usualmente immagazzinato nei chips della ROM e non viene mai "dimenticato"; ogni volta che il computer viene acceso questi programmi sono già pronti in memoria e funzionanti. Ma questo è solo per una questione di comodità, è importante comprendere che il computer potrebbe essere composto esclusivamente da memoria RAM ed in tal caso (SAREBBE) realmente necessario caricare ogni volta il sistema operativo ed il linguaggio di programmazione dal disco o digitarli all'accensione del computer come tutti gli altri programmi. In realtà questo tipo di configurazione è possibile con il C64 che ha 64 Kbyte di memoria RAM a cui si sovrappongono aree ROM (questo è il motivo per cui ce ne sono solo 39 Kbyte disponibili all'utente). Con qualche POKE adeguata è possibile disabilitare la ROM e ottenere 64 Kbyte completamente disponibili ma i programmi, allora, dovranno tenere conto di tutte le operazioni normalmente gestite dal Sistema Operativo che si è cancellato. In questo caso anche il BASIC è stato disinserito con le ROM, la programmazione potrà dunque essere realizzata solo in linguaggio macchina, solo i programmatori più avanzati possono farsi carico di un compito così complesso.

Così se il software di base è analogo al sistema nervoso involontario, è facile immaginare che il software applicativo paragonato al pensiero cosciente. Liberato dalle operazioni di routine con l'uso di un adeguato sistema operativo, il computer può dedicare tutte le sue risorse all'esecuzione delle più svariate applicazioni che l'utente possa immaginare, dai più sofisticati "data base" agli "invasori spaziali". L'analogia tra computer e cervello umano non deve andare più lontano di quanto sia necessario, è opportuno, comunque, ricordare che, come il cervello, il computer non può apparire più intelligente di quello che ha in "mente".

Il bus seriale collega il VIC ed il Commodore 64 alle loro più importanti periferiche, in particolare a disco e cassetta. Il funzionamento di questa interfaccia è stato fonte di incertezza per molti, pur sapendo che è in qualche modo simile al bus IEEE-488 che è utilizzato sul pet e sugli altri CBM. Per chiunque sia interessato ai problemi di interfacciamento questo articolo chiarirà finalmente il mistero.

Il bus seriale del VIC e del C64

di Jim Butterfield
trad. e adatt. F. Stella

Concetti di base

Per capire il funzionamento di questa interfaccia è necessario chiarire alcuni concetti, poi, per coloro che sono interessati ad approfondire l'argomento, entreremo nei dettagli più tecnici. Il BUS seriale, come l'IEEE, ha due modi di funzionamento; SELECT MODE, in cui il computer chiama tutti i dispositivi e chiede ad uno in particolare di restare collegato dopo la chiamata ("Rossi, le dispiace rimanere nel mio ufficio dopo la riunione?"); ed il DATA MODE, nel quale viene trasmessa l'informazione vera e propria ("Rossi, ho deciso di aumentarle lo stipendio!"). Il modo select è innescato dall'uso di una speciale linea di controllo chiamata ATN (cioè "attention"). In questa fase, è possibile richiamare qualsiasi dispositivo periferico, ma ciò non è sufficiente per passare alla trasmissione dei dati; è possibile che vi siano diversi file su disco aperti, alcuni per la lettura ed altri per la scrittura, cosicché, scegliendo come periferica il disco, device n. 8, è ancora necessario specificare quale "parte" del disco si vuole raggiungere: il canale 3, il canale 15 o quale altro sia. Per fare questo si usa l'indirizzo secondario che segnala una particolare sezione di un "device" richiamato o un particolare modo operativo. Altra informazione da inviare in questa fase preliminare è il nome del file che si intende aprire. Lo schema base, comunque, è:

richiamo un dispositivo e poi vi trasmetto o ricevo informazioni; alla fine lo spengo.

Tutti i dispositivi sono collegati ma solo quello prescelto "ascolta" e "parla" quando richiesto.

Principi tecnici

Se siete tra quelli a cui le parole volt, segnale e simili dicono poco, quello che segue potrà essere, per voi, di difficile comprensione ma è comunque indispensabile parlare degli aspetti tecnici del funzionamento del BUS. Ci sono solo due fili su cui si trasferiscono i dati, sono la linea CLOCK e la linea DATA; ce ne sono in realtà altri ma sono usati per la realizzazione di controlli. Il cavo che parte dal computer è in contatto con tutte le periferiche ed i fili non sono a "senso unico", ogni dispositivo può mettere a terra una linea e tutti gli altri possono rilevarlo; questo è il segreto del funzionamento: ogni filo serve come linea di segnalazione comune. Quando nessun dispositivo mette a terra la linea il voltaggio sale a circa 5 volt; questa viene definita condizione di falso logico del connettore (false). La condizione di "vero" (true) si ha quando il voltaggio scende a zero; se uno o più dispositivi segnalano un "true" sulla linea comune l'effetto è lo stesso. Sommariamente si possono delineare due regole logiche:

- Una linea diventa "true" se uno o più dispositivi lo segnalano.
- Una linea diventa "false" solo se tutti i dispositivi sono staccati.

È opportuno ricordare che vi sono diverse linee ma che quelle importanti ai fini della trasmissione dei dati sono solo la linea CLOCK e la linea DATA; vediamo come lavorano.

Trasmissione: fase zero

Cosa succede quando un carattere sta per essere trasmesso? Sia la linea CLOCK che la DATA sono mantenute basse allo stato "true"; con un tester non è possibile specificare chi lo sta facendo. Il dispositivo che deve trasmettere mantiene la linea CLOCK e ciascuno di quelli che potrebbero ricevere la linea DATA. Ogni segnale potrebbe essere interpretato come un "Sono qui!".

Fase uno: ready to send

Prima o poi, l'emittente dovrà inviare un carattere; una volta pronto rilascerà la linea CLOCK che passerà a "false" come per avvertire: "Sono pronto ad inviare un carattere". Il ricevente dovrà accorgersene e rispondere appena possibile, non c'è limite di tempo. Ad esempio, una stampante occupata su una linea o una unità disco con un lavoro di formattazione in corso possonno ritardare per un tempo relativamente lungo.

Fase due: ready for data

Quando il ricevente è pronto rilascerà a sua volta la linea DATA. Nel caso vi fossero più riceventi, la linea DATA assumerà il valore logico "false" solo quando tutti avranno compiuto questa operazione e saranno pronti. Ora l'emittente potrà riportare la linea CLOCK a "true" in meno di 200 nanosecondi (usualmente 60) oppure non rispondere. Se la risposta non arriva si tratta di un segnale EOI.

EOI

Se il READY FOR DATA non riceve risposta entro 200 nanosecondi significa che l'emittente sta segnalando che il prossimo è l'ultimo carattere (EOI = End OF — Indicator). Se si tratta di un file sequenziale su disco, è la fine del file; se di un record relativo dell'ultimo carattere del record. Il ricevente deve dare conferma tenendo la linea DATA "true" per almeno 60 nanosecondi. L'emittente riprenderà quindi ponendo la sua linea (CLOCK) "true" e proseguendo la trasmissione in modo normale.

Fase 3: invio dei bit

Ci sono otto bit che devono essere trasmessi senza "handshake" (= stretta di mano), cioè senza che il ricevente debba ogni volta confermare la sua attenzione. L'emittente controlla ora entrambe le linee: CLOCK "true" e DATA "false" sulla linea DATA vengono impostati i bit, uno alla volta, a partire dal

meno significativo; ad esempio il carattere ASCII "?", che è rappresentato dal numero binario 00011111, sarà trasmesso a cominciare dagli uno per finire con gli zeri. La linea DATA è impostata a "true" se il digit è uno e a "false" in caso contrario. Una volta pronto il bit da trasmettere la linea clock dà un "data ready" portandosi su "false"; immediatamente dopo le due linee sono mantenute per un minimo di 20 nanosecondi allo stato in cui si trovano per dare il tempo di leggere. Quest'ultimo tempo deve essere incrementato a 60 nanosecondi se il ricevente è il C 64 perché il suo circuito video può interrompere l'operazione fino a 42 nanosecondi ogni volta e senza questo tempo maggiorato si rischierebbe di perdere completamente alcuni bit. Il ricevente in questa fase svolge un ruolo passivo: non invia nulla, aspetta; appena vede la linea CLOCK rilasciata a false prende il valore dall'altra e lo immagazzina.
Concluso l'invio del primo bit l'emittente porta la CLOCK a "true" e rilascia la DATA a "false" preparandosi all'invio dei seguenti.

Fase 4: handshake

Concluso l'invio dell'ottavo bit tocca al ricevente dare conferma mettendo la linea DATA (true) entro un millisecondo (mille nanosecondi) altrimenti il computer saprà che qualcosa non ha funzionato e provvederà al conseguente messaggio di errore.

Fase 5: pronti per un nuovo ciclo

A questo punto il ciclo è concluso e tutto è pronto per un nuovo carattere se non c'è stata una segnalazione di EOI nel giro appena concluso; in tal caso sia l'uno che l'altro dispositivo lasciano le linee a "false" e la trasmissione si conclude definitivamente.

Attenzione!

Tutto quello che abbiamo visto è molto chiaro nel caso di una trasmissione già in corso, ma come si fa a determinare chi parla e chi ascolta? Si usa una linea speciale, chiamata ATN. Normalmente il computer è l'unico dispositivo che può portare la linea ATN a "true". Quando lo fa tutti gli altri dispositivi interrompono ogni attività in corso di esecuzione e diventano riceventi. Durante una fase ATN vengono inviati caratteri con la stessa procedura vista prima, ma non sono dati, sono i comandi TALK, LISTEN (rispettivamente trasmessi, ricevi), UNTALK, UNLISTEN che segnalano ad un dispositivo ben preciso cosa diventare o cosa cessare di essere. Questi comandi vanno a tutti i dispositivi collegati e tutti li riconoscono sebbene solo quelli richiamati li eseguano. Dopo il comando spesso segue un indirizzo secondario e dopo che ATN è stata rilasciata anche un

REM:HW

Il bus seriale del VIC e del C64

nome di file. Vediamo un esempio che può facilitare la comprensione del tipo di comunicazioni che avvengono tra l'emittente ed il ricevente; immaginiamo l'apertura di un file sequenziale su disco chiamato "xx": la sequenza da inviare è ATN on: DEVICE — 8 LISTEN: SECONDARY — ADRESS — 2 — OPEN: quando poi la ATN viene rilasciata il computer rimane in attesa come emittente ed invia i caratteri X — X — virgola — S — virgola — W: dove la W è preceduta da un EOI. Subito dopo la linea ATN è riportata a "true" e viene inviato un DIVICE — 8 — UNLISTEN. Il file rimane aperto e quando il computer vorrà inviare dati trasmetterà la nuova sequenza ATN on: DEIVCE — 8 — LISTEN: SECONDARY — ADDRESS — 2 — DATA, rilascerà di nuovo ATN e procederà come abbiamo già visto. Dopo la trasmissione di tutti i dati previsti la nuova sequenza DEVICE — 8 — LISTEN: SECONDARY — ADDRESS 2 — CLOSE chiuderà il file e l'operazione in corso. La linea ATN ha priorità su tutte le operazioni in corso cosicché se, ad esempio, un file su disco contiene più informazioni di quante ne servano, una volta "soddisfatto", il computer può bloccare la trasmissione anche senza che questa sia conclusa (avvenuta segnalazione EOI) per mezzo della ATN.

Sequenze ATN

Quando la linea ATN è "true" tutti interrompono quindi le loro attività. Il processore mette a "true" anche la linea CLOCK (sta per inviare i comandi), cosicché è difficile notare che gli altri dispositivi la rilasciano; per la linea DATA invece avviene il contrario, il computer la rilascia e gli altri dispositivi la riportano "true" entro un millisecondo per evitare che il processore finisca col segnalare un "DEVICE NOT PRESENT ERROR". Se tutto procede bene la trasmissione continua esattamente come descritto prima, sebbene il computer stia inviando comandi anziché dati. Tutti i dispositivi periferici ricevono i comandi ma solo quelli interessati passano all'azione. Questa è una situazione molto curiosa in quanto è possibile inviare comandi a periferiche inesistenti (ad. es. OPEN6,6) ed il computer non potrà accorgersene in quanto riceverà un valido "handshake" dagli altri dispositivi in linea. Quando però tenterà di inviare dei dati si troverà a parlare con il vuoto, in quanto tutti gli altri lasciano le linee non appena il segnale ATN è cessato.

Conclusioni

Una sequenza inusuale avviene quando il computer vuole assegnare alla periferica il ruolo di emittente. Questo accade subito dopo un comando di TALK, ed appena la linea ATN è rilasciata il dispositivo selezionato si trova quindi in posizione d'ascolto. Era ricevente durante l'ATN in quanto il computer trasmetteva; così ci si trova in una situazione logica opposta a quella desiderata: il dispositivo mantiene "true" la linea DATA ed il computer la linea CLOCK. Per aggirare questo ostacolo si verifica questa sequenza: il computer si "accorge del problema" e mette la linea DATA a "true" (anche se già lo è) e contemporaneamente rilascia la linea CLOCK; il dispositivo era in attesa di questo passo e appena vede la linea CLOCK scendere, rilascia la linea DATA (che comunque non cambia stato perché è stata rilevata dal computer) e lascia la linea CLOCK. Ora siamo nella posizione esatta di partenza con l'emittente (il dispositivo) che mantiene la linea CLOCK ed il ricevente (il computer) che mantiene la linea DATA. Il computer aspetta che si configuri questo stato prima di approntarsi a ricevere dati. Le sequenze logiche sono ripristinate. In ogni caso è però difficile rilevarle con un voltmetro od un oscilloscopio in quanto non si può mai determinare chi stia mantenendo una linea e chi no.
Paragonandolo all'IEEE-488 il bus seriale del VIC e del C 64, pur avendo gli stessi comandi TALK e LISTEN e gli stessi indirizzi secondari, ha meno linee di handshake e la velocità leggermente ridotta; sono mantenuti comunque gli stessi principi di base.

Piccola guida per l'input dei programmi

Molti dei programmi listati da **SUPERVIC** contengono particolari caratteri di controllo (controllo cursore, tasti colore, video-inverse, etc.).
Per rendere chiaro ciò che dev'essere battuto quando è necessario inserire uno dei suddetti caratteri sono state stabilite le seguenti convenzioni.
Generalmente i listati per VIC 20 e C64 contengono delle «parole» racchiuse tra parentesi graffe { }; tali parole rappresentano particolari caratteri di controllo: **{GIÙ}** significa premere il simbolo del cursore verso il basso. **{5 SPAZI}** vuol dire battere 5 volte la barra-spazio.
Per indicare che un tasto dev'essere «shiftato», cioè premuto insieme al tasto SHIFT, nel listato il simbolo di quel tasto sarà sottolineato.
Per esempio <u>S</u> significa che dev'essere premuto il tasto S mentre è abbassato il tasto SHIFT.
La manovra farà apparire sullo schermo un piccolo «cuore».
Trovando un simbolo sottolineato chiuso tra parentesi graffe (esempio **{10 <u>N</u>}**) dovrà essere interpretato come «premere il carattere indicato per il numero di volte che lo precede nelle parentesi» nel nostro esempio premere 10 volte il tasto N «shiftato».
Se il tasto da premere è indicato tra le parentesi [( )] vuol dire che lo stesso dovrà essere premuto mentre è premuto il tasto «Commodore» (il tasto «Commodore» è quello posto nell'angolo in basso a sinistra).
Se il tasto scritto tra [( )] è preceduto da un numero, ciò significa che il tasto dev'essere premuto per il numero di volte indicato.
Raramente si potrà trovare un carattere alfabetico solitario racchiuso tra parentesi graffe. Tale carattere sul C64 può essere battuto mentre è premuto il tasto CTRL.
Ad esempio **{A}** sta ad indicare la sequenza CTRL-A.
A proposito del «modo virgolette» è nota la possibilità di muovere il cursore sullo schermo con i tasti CRSR. Ogni tanto i programmatori desiderano muovere il cursore durante l'esecuzione del programma. È per questo motivo che nei programmi si troveranno dei **{SIN}**, **{HOME}** e **{BLU}**.
L'unico modo perché il calcolatore distingua il comando diretto da quello inserito nel programma è il «modo virgolette». Premendo il tasto «virgolette» (tasti SHIFT 2) il calcolatore si predispone appunto in «modo virgolette».
Battendo un carattere qualsiasi cercando poi di correggerlo muovendo il cursore a sinistra, si otterrà solo un tratto verticale in un quadratino inverso. Questo è infatti il simbolo del cursore a sinistra nel «modo virgolette».
L'unico comando di editing non utilizzabile all'interno di un programma è il DEL. Battendo nuovamente le «virgolette» il calcolatore lascia il «modo virgolette». Si può accedere al «modo virgolette» quando inserite spazi nella linea.
In ogni caso, il sistema più semplice per uscire dal «modo virgolette» è quello di premere il tasto RETURN.
Utilizzare la tabella che segue quando è necessario inserire comandi relativi al cursore e ai tasti colore.

Quando leggete:	Premete:	Vedrete:	Quando leggete:	Premete:	Vedrete:	Quando leggete:	Premete:	Vedrete:
{CLR}	SHIFT CLR/HOME		{CYN}	CTRL 4		[<7>]	C= 7	
{HOME}	CLR/HOME		{PUR}	CTRL 5		[<8>]	C= 8	
{SU}	SHIFT ⇑ CRSR ⇓		{GRN}	CTRL 6		{F1}	f1	
{GIU'}	⇑ CRSR ⇓		{BLU}	CTRL 7		{F2}	f2	
{SIN}	SHIFT ⇐ CRSR ⇒		{YEL}	CTRL 8		{F3}	f3	
{DES}	⇐ CRSR ⇒		[<1>]	C= 1		{F4}	f4	
{RVS}	CTRL 9		[<2>]	C= 2		{F5}	f5	
{OFF}	CTRL 0		[<3>]	C= 3		{F6}	f6	
{BLK}	CTRL 1		[<4>]	C= 4		{F7}	f7	
{WHT}	CTRL 2		[<5>]	C= 5		{F8}	f8	
{RED}	CTRL 3		[<6>]	C= 6				

OK-LIST (mai più errori di ortografia!)



di Charles Brannon
trad. e adatt. R. Paolillo

Tutti i listati pubblicati da SUPER VIC sono ottenuti mediante un calcolatore ed una stampante direttamente da versioni perfettamente funzionanti e sottoposte a prove esaustive dei programmi, tuttavia qualche problema può sempre sorgere. Per evitare inutili frustrazioni, viene proposto l'OK-LIST, un modo rivoluzionario di introdurre i programmi che vi avverte all'istante se avete commesso un errore di battitura.

Sappiamo tutti quanto sia difficile digitare correttamente un programma fin dalla prima volta. Errori di battitura apparentemente banali possono provocare i temuti messaggi ERROR, o provocare anche il blocco completo del sistema (la tastiera non reagisce alla pressione dei tasti RUN/STOP e RESTORE). Generalmente l'unico modo di ovviare a simili blocchi totali consiste nel far ripartire il calcolatore mediante spegnimento e riaccensione, cancellando, però, la memoria e tutto il vostro lavoro di battitura.
Anche quando riuscite a localizzare ed a correggere le righe errate, sembrano sempre esserci altri errori in agguato nelle centinaia di righe che compongono il programma. Talvolta viene la voglia di arrendersi.

Errori elusivi

Alcuni errori sono quasi impossibili da rilevare, specialmente per i principianti che sanno poco o nulla di programmazione. Ad esempio, sapete identificare l'errore presente in questa riga?

100 PRINTRIGHT$ ("00" + MID$ (STR$ (V),2,3)

Ecco come dovrebbe essere:

100 PRINTRIGHT$ ("00" + MID$ (STR$ (V),2),3)

Avete notato la differenza? Mancava una parentesi chiusa dopo il numero 2. Una parentesi aperta deve avere sempre una parentesi chiusa corrispondente. Se sommate tutte le parentesi presenti in una istruzione, dovete ottenere un egual numero di parentesi aperte e chiuse.

Un sogno irrealizzabile?

Il punto di forza dei calcolatori è che essi eccellono nei compiti noiosi e che richiedono precisione. Quindi, perché non utilizzare il vostro calcolatore per controllare i testi da voi digitati? È un sogno irrealizzabile?
Non con l'**OK-LIST**! Annidato all'interno del vostro VIC 20 o C 64, il programma controlla automaticamente ogni riga battuta. Mostra un numero in cima al vostro schermo. Questo numero, chiamato "checksum", corrisponde all'ultima riga da voi introdotta. Rappresenta la somma del valore ASCII di ogni carattere presente nella riga. Un numero corrispondente nel listato del programma pubblicato vi permette di confrontare il valore numerico di controllo con il "checksum" mostrato da **OK-LIST**. Uno sguardo è sufficiente per avere la conferma di aver digitato la riga correttamente.
OK-LIST è un breve programma in linguaggio macchina residente in un area di memoria relativamente sicura, il buffer del registratore. Rimarrà in tale zona finché non spegnete il calcolatore o manderete in esecuzione un altro programma che utilizza il buffer del registratore. Caricare o salvare programmi BASIC da e su disco, non avrà alcuna influenza sul programma in questione.

Come inserire l'OK-LIST

Battete il programma che vale sia per il VIC 20 sia per il C64. C'è una sola piccola trappola: il programma non può controllare se stesso, quindi porre particolare attenzione nella sua battitura. Dato che si tratta di un programma in linguaggio macchina, siate estremamente attenti. Fate attenzione a non battere delle virgole in più, la lettera O invece di zero, e controllate ogni valore numerico accuratamente. Fortunamente, **OK-LIST** è un programma breve, quindi non dovreste avere eccessivi problemi.
Dopo aver introdotto **OK-LIST**, registratelo su disco o nastro per lo meno un paio di volte prima di mandarlo in esecuzione la prima volta. Se avete

commesso errori di ortografia nel programma, ciò può provocare un blocco completo del sistema quando lo eseguite per la prima volta. Tenendone una copia registrata a portata di mano, avrete la possibilità di ricaricarlo ed andare a caccia dell'errore. Inoltre, avrete bisogno di una copia di **OK-LIST** poiché sarà utilizzato sempre più spesso, ogni volta che copierete un programma dalla nostra rivista.
Quando manderete in esecuzione **OK-LIST**, il programma verrà copiato in memoria dove rimarrà al sicuro. Premete quindi il tasto RETURN sulla riga dove il cursore si trova, in modo da attivare il programma. Se occorresse riattivarlo, limitatevi a battere **SYS886** seguito da RETURN.

Come si usa OK-LIST

Ora, vediamo come funziona. Listate il programma che state digitando, muovete il cursore in modo da sovrapporvi ad una delle righe, e premete RETURN. Se avete digitato **OK-LIST** in maniera precisa, in cima al vostro schermo apparirà un valore numerico.

10A = 1:B = 72:PRINT "SCORE = ";
SC:rem 199

NON DIGITATE il commento dopo i due punti, rappresenta il valore di "checksum").
Provate a modificare la riga, premete RETURN, e notare che il numero è cambiato. Tutti i listati per VIC 20 e C 64 che appariranno sulla nostra rivista avranno un numero posto alla fine di ogni riga, ad esempio ":rem 123".

Non inserite questo numero nella linea d'istruzione!

Serve solo per vostra conoscenza. Il comando "rem" è utilizzato per rendere il valore numerico innocuo, nel caso in cui qualcuno lo inserisca comunque. Tuttavia, ciò occuperebbe inutile spazio in memoria e farebbe sí che il numero di controllo in cima allo schermo risultasse differente, pur nel caso in cui la riga sia inserita correttamente. Limitatevi a battere ogni riga (**senza il valore numerico di controllo stampato accanto**), e controllate che il numero mostrato in cima allo schermo coincida con il valore numerico di controllo del listato. Se coincidono, proseguite battendo immediatamente la riga successiva. Diversamente, c'è un errore. Potete correggere la linea invece di aspettare e trovare l'errore quando mandate in esecuzione il programma.
OK-LIST non è pignolo per quel che riguarda gli spazi bianchi. Non farà caso a spazi in più o in meno. Ciò torna a vostro vantaggio, dal momento che la spaziatura non è generalmente importante. Ma occasionalmente una spaziatura appropriata è importante, quindi siate particolarmente accurati per quel che riguarda gli spazi, dal momento che il programma rileverà praticamente qualsiasi altro errore.
C'è un'altra cosa cui dovete badare: se utilizzate i comandi abbreviati per digitare la riga, il valore numerico di controllo non coinciderà. Ma esiste un modo per far si che **OK-LIST** controlli la riga. Dopo averla digitata, listatela. Ciò elimina le forme abbreviate dei comandi. Muovete quindi il cursore fino a portarlo in corrispondenza della riga e premete RETURN. Ora il valore numerico deve corrispondere. Potete controllare in questo modo interi gruppi di righe.
Quando ne avete abbastanza potete disattivare **OK-LIST** premendo RUN/STOP-RESTORE (tenete premuto il tasto RUN/STOP e premete RESTORE). Se ne avete ancora bisogno, battete **SYS886**. Sarà quindi ancora disponibile per operare come vostro supporto personale all'introduzione dei programmi.
Speriamo che **OK-LIST** renda l'introduzione dei programmi più veloce e più facile, e che non dobbiate affrontare più frustranti messaggi di errore.

LOAD

OK-LIST (mai più errori di ortografia!)

Istruzioni speciali per il salvataggio su nastro

Quando avete terminato di copiare un listato, dovete disattivare **OK-LIST** prima di registrare (SAVE) il programma su nastro. Disattivate **OK-LIST** premendo RUN/STOP-RESTORE (tenete premuto il tasto RUN/STOP e premete rapidamente RESTORE). Questa serie di operazioni non è necessaria per salvataggi su disco, ma dovete disattivare assolutamente **OK-LIST** nel modo descritto prima di registrare su nastro.
Il salvataggio su nastro cancella **OK-LIST** dalla memoria, quindi è necessario caricarlo (LOAD) e mandarlo in esecuzione (RUN) nuovamente se volete copiare un altro listato. Registrazioni su disco non cancellano il programma.

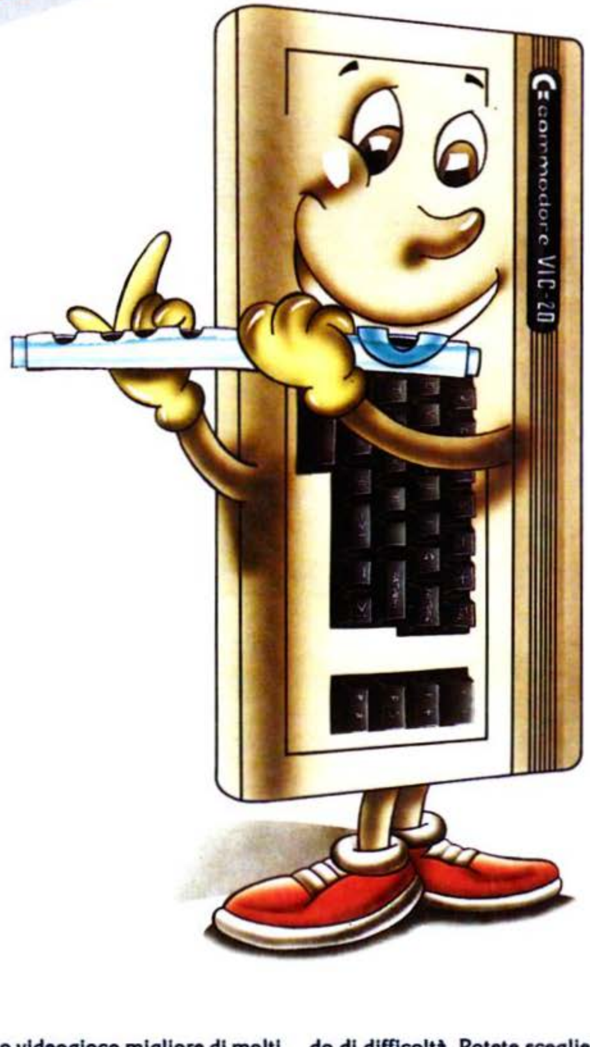
OK-LIST versione per VIC 20 e C 64

```
100 PRINT"{CLR}ATTENDERE PREGO..."
105 FORI=886TO1018:READA:CK=CK+A:POKEI,A:
    NEXT
110 IF CK<>17539 THEN PRINT"{GIU'}ERRORE
    NELLE ISTRUZIONI DATA":END
120 SYS886:PRINT"{CLR}{ 2 GIU'}OK-LIST AT
    TIVATO.":NEW
886 DATA 173,036,003,201,150,208
892 DATA 001,096,141,151,003,173
898 DATA 037,003,141,152,003,169
904 DATA 150,141,036,003,169,003
910 DATA 141,037,003,169,000,133
916 DATA 254,096,032,087,241,133
922 DATA 251,134,252,132,253,008
928 DATA 201,013,240,017,201,032
934 DATA 240,005,024,101,254,133
940 DATA 254,165,251,166,252,164
946 DATA 253,040,096,169,013,032
952 DATA 210,255,165,214,141,251
958 DATA 003,206,251,003,169,000
964 DATA 133,216,169,019,032,210
970 DATA 255,169,018,032,210,255
976 DATA 169,058,032,210,255,166
982 DATA 254,169,000,133,254,172
988 DATA 151,003,192,087,208,006
994 DATA 032,205,189,076,235,003
1000 DATA 032,205,221,169,032,032
1006 DATA 210,255,032,210,255,173
1012 DATA 251,003,133,214,076,173
1018 DATA 003
```

Re-Beep

di Robert L. Lykins
trad. e adatt. U. Barzaghi

Re-Beep è un gioco simile a "Simon", ideato originariamente per il VIC 20 privo di espansione di memoria. In seguito è stata aggiunta una trascrizione per C 64.



"Re-Beep" è un gioco per VIC 20 e C 64 che vi terrà letteralmente con le orecchie ben aperte. Viene richiesta una capacità di concentrazione sempre maggiore per migliorare il vostro punteggio. Dovete ripetere la sequenza di suoni, che si va allungando di mossa in mossa, fornita dal calcolatore, premendo i tasti speciali di funzione nell'ordine esatto.
Se ripetete correttamente la sequenza, il vostro punteggio aumenta di una unità. (Si ottiene un punto per ogni suono che fa parte della stringa musicale proposta). Quindi il calcolatore aggiunge una nota alla vecchia stringa, e voi dovete tentare di riprodurre la sequenza più lunga. Il calcolatore inizia con una nota, il che è abbastanza facile. Ben presto, tuttavia, udrete una inebriante serie di suoni. Riuscirete a resistere fino ai 127 suoni che il calcolatore può emettere? Probabilmente no. Io mi considero fortunato se riesco a far segnare una ventina di punti. Per coloro che sono dotati di ottima memoria, tuttavia, c'è la possibilità di aumentare le capacità del calcolatore di produrre stringhe sonore, fino a portarle ad una lunghezza di 255 note. Ne parleremo più avanti.
Quando la vostra memoria vi tradisce, non tutto è perduto. Il calcolatore suonerà un cicalino, vi dirà di riprovare, quindi ripeterà per voi la sequenza musicale. Chi dice che un calcolatore non sa essere umano? Se, nonostante lo shock di aver fallito la vostra prima prova, riuscite a riguadagnare la vostra concentrazione ed a suonare opportunamente la sequenza, il gioco continua. Se non ce la fate nemmeno la seconda volta, sarete "sonoramente" ripresi ed il calcolatore inizierà un'altra sequenza.

Caratteristiche d'avanguardia

Parecchie caratteristiche incorporate in questo gioco simile a Simon, rendono questo videogioco migliore di molti altri. Innanzi tutto ricorda il vostro record di punteggio finché non cancellate il programma o non spegnete il calcolatore. Alcune stringhe sono più facili da ricordare di altre, e vi accorgerete di non riuscire ad offrire un rendimento costante. Il calcolatore registra e mostra il record da voi raggiunto, in modo da fornirvi una meta cui tendere nelle "manches" successive.
Una seconda caratteristica è rappresentata dalla facoltà di scegliere il grado di difficoltà. Potete scegliere fra tre livelli che regolano la velocità a cui le note vengono suonate. I principianti e i bambini di pochi anni faranno probabilmente del loro meglio sul primo livello. Il terzo livello richiede prontezza di spirito, mentre il secondo è un buon compromesso tra i due. Non è tanto lento da essere noioso, nè tanto veloce da farvi rimanere a bocca aperta e dire, "E questo cos'era?"
La miglior caratteristica di questo programma è forse rappresentata dal fatto

LOAD

Re-Beep

di poter giocare in due persone. La maggior parte dei giochi per calcolatore sembrano essere ideati per un solo giocatore. Con Re-Beep, potete scegliere di giocare contro un'altra persona. In questo caso, il calcolatore chiederà il nome di ciascun giocatore e fornirà un tabellone segnapunti separato per entrambi. Anche i record verranno mostrati separatamente.
Il primo giocatore inizia, ed il numero corrispondente al suo segnapunti rimane acceso finché il giocatore non perde il turno. Quindi il calcolatore accende il numero corrispondente al secondo segnapunti e dà inizio ad una nuova sequenza per il secondo giocatore. Quando le capacità di concentrazione del secondo giocatore vengono meno, il gioco ritorna al primo giocatore ed al suo segnapunti. Il gioco viene così alternato tra i due segnapunti. Se una sola persona vuole giocare, il calcolatore salterà la richiesta dei nomi, mostrerà solo il segnapunti superiore, e non accenderà il numero del segnapunti. Ciò permette di concentrare l'attenzione sul proprio compito. Si possono formare due squadre, se più di due persone desiderano giocare. Ogni segnapunti ha a disposizione 12 caratteri per il nome o i nomi, ma se ne vengono usati due, non devono essere separati da virgola, a causa della caratteristica dell'INPUT del calcolatore di trascurare qualsiasi cosa segua una virgola. Una barra (/), comunque, dovrebbe essere adatta allo scopo.



Come funziona la versione VIC 20

Questo programma sfrutta una tecnica nota come "concatenazione di stringhe" l'unione di due stringhe. (Una stringa è una serie di caratteri). In questo caso, STR$ (R) viene aggiunto a **S$**, la stringa contenente le informazioni che il calcolatore utilizza per suonare la serie di note. **R** è un numero casuale d 1 a 4 inclusi, corrispondente ad uno dei quattro tasti di funzione, che viene scelto in riga 230. È necessario convertirlo in una stringa tramite una istruzione STR$ (R) in modo che possa essere utilizzato in una sequenza numerica.
Ad esempio, se **S = 0, S$ = " "**, ed **R** è prima uguale a 3, quindi a 4, allora:

Somma Algebrica

S = S + R
S = 0 + 3 = 3
S = 3 + 4 = 7

Concatenazione di stringhe

S$ = S$ + STR$ (R)
S$ = " " + "3" = "3"
S$ = "3" + "4" = "34"

In questo esempio, R corrisponde inizialmente al tasto di funzione **F5 (3)**, quindi al tasto **F7 (4)**. Sommare tre e quattro algebricamente dà come risultato 7, una sola cifra che non corrisponde ad un tasto. Ma concatenando le stringhe come in riga 240 si ottiene una sequenza dei valori **R** selezionati che corrispondono effettivamente a dei tasti di funzione e a delle note.
Utilizzando le righe 260 e 300, il calcolatore suona una serie di note e illumina i rettangoli verdi corrispondenti ai tasti di funzione.
Ciò viene realizzato tramite la conversione in un valore numerico **(Q)** di ciascun numero contenuto nella stringa **S$** in riga 270. La riga 290 devia il corso del programma verso una delle quattro subroutine che emettono i suoni ed illuminano i rettangoli, in base al valore della variabile **Q**.
Una maneggevole istruzione ON GOSUB rimpiazza quattro istruzioni IF...THEN. Se **Q = 1**, il programma raggiunge (GOSUB) la riga corrispondente al primo valore numerico della riga, 480. Se **Q = 2**, raggiunge (GOSUB) la seconda, ecc..
l'istruzione NEXT in riga 300 ripete il procedimento se la riga 260 indica che ci sono altri caratteri (in base alla lunghezza LEN della stringa **S$**). Forse noterete nell'esempio di concatenazione delle stringhe che i valori numerici sono preceduti da spazi bianchi. Ciò avviene poiché la istruzione STR$ (R) inserisce uno spazio bianco davanti ad ogni valore numerico, ed un controllo tramite la funzione LEN lo rivela: LEN (STR$ (4)) dà un valore uguale a due. È quindi necessario che il ciclo FOR-NEXT che inizia in riga 260, inizia con **2** ed ha un passo (STEP) di **2**, così che la riga 270 non legga gli spazi bianchi. Se desiderate rallentare lo svolgimento del gioco, potete iniziare il ciclo con **1** e eliminare la parte STEP dell'istruzione. Ciò farà si che **Q** assuma valore **0** tutte le altre volte, la qual cosa non provocherà alcun suono. Lasciando invece la parte STEP si avrà una riga 270 che legge solo spazi bianchi, ed una riga 290 che non eseguirà mai sottoprogrammi di emissione delle note.
Dopo che tutte le note contenute nella stringa **S$** sono state suonate, le righe 310-350 acquisiscono la risposta del giocatore tramite i tasti di funzione, suonano le note opportune utilizzando le stesse subroutine, e costruiscono la stringa **F$**. La riga 370 confronta la lunghezza della stringa del giocatore **(F$)** e quella della stringa del calcola-

tore (**S$**). Se hanno la stessa lunghezza (LEN), significa che il giocatore ha suonato tante note quante quelle suonate dal calcolatore e vengono fatte ulteriori analisi. Altrimenti, il programma ritorna alla riga 310 per acquisire altri tasti premuti dal giocatore.
La riga 380 stabilisce se la sequenza del giocatore è corretta oppure no. Se non lo è, permette al giocatore di riprovare se **E = 0**, quindi cambia il valore di **E** a **1**.
Ciò fa sì che il programma salti la routine "Riprova", nel caso in cui il giocatore sbagli anche la seconda volta.
La riga 400 percepisce tutto ciò e devia il corso del programma al sottoprogramma che si occupa di chiudere il gioco. Se il programma riesce a superare le righe 380 e 400 senza essere deviato, il giocatore ha correttamente suonato la sequenza sonora, ed il programma accede alla routine che attribuisce il punteggio. Quindi il programma aggiunge un altro valore numerico alla stringa **S$** e allunga la sequenza sonora. La procedura si ripete.

Per i seguaci di Pico della Mirandola...

Permettendo alla funzione STR$ di inserire spazi bianchi nelle stringhe **S$** e **F$**, si spreca in pratica metà della capacità di memoria disponibile per la sequenza sonora, limitandola a 127 note, quantunque questa quantità dovrebbe dimostrarsi più che sufficiente per la maggior parte dei giocatori. Forse alcuni lettori hanno effettivamente capacità mnemoniche straordinarie. In questo caso, i seguenti suggerimenti aumenteranno le difficoltà della sfida insita nel gioco. O, perlomeno, illustrano ciò che si può fare — e può essere utile in altre applicazioni — per risolvere il problema costituito dallo spreco di memoria.
Per riempire **S$** e **F$** con numeri utili (ed ottenere una capacità di 255 note), devono essere aggiunti solo i valori numerici e non gli spazi bianchi. Ciò si ottiene utilizzando la funzione RIGHT$ come viene illustrata più sotto alle righe 240 e 320-350. In questo caso, il carattere più a destra ricavato dalla istruzione **STR$**, vale a dire il valore numerico, viene aggiunto alla stringa musicale invece dell'intero risultato della funzione **STR$**.

Per ottenere 255 note cambiate le seguenti righe:

```
240  S$ = S$ + RIGHT$  (STR$(R),  1)
260 FORL = 1TOLEN (S$)
320  IFZ$ = ''(F1)''THENGOSUB
480 F$ = F$ + RIGHT$(STR$(1),1)
330  IFZ$ = ''(F3)''THENGOSUB
540 F$ = F$ + RIGHT$(STR$(2),1)
340  IFZ$ = ''(F5)''THENGOSUB
600 F$ = F$ + RIGHT$(STR$(3),1)
350  IFZ$ = ''(F7)''THENGOSUB
660 F$ = F$ + RIGHT$(STR$(4),1)
```


Re-Beep

Altre tecniche

Il livello di gioco dipende dal valore della variabile **LV** (da **1** a **3**), che viene acquisito da tastiera all'inizio del programma (riga **50**). Alcune righe all'interno del programma utilizzano dei cicli di temporizzazione che contengono la variabile **LV**. **LV** viene elevata al quadrato e viene quindi usata per dividere la lunghezza del ciclo di attesa. Se si gioca al primo livello, il numero viene diviso per uno quindi non cambia. Se **LV = 2**, il numero viene diviso per quattro (**LV²**), riducendo sostanzialmente il ciclo di attesa. E al terzo livello, il valore numerico viene diviso per nove.
I cicli vengono utilizzati anche per stampare parte delle uscite video. Il titolo viene generato in questo modo dalla riga 40. Differenti colori vengono ottenuti con una istruzione di POKE646,L. Le righe 150 e 160 stampano i numeri dei tasti di funzione **F** e li accoppiano ai rettangoli verdi. Utilizzando questa tecnica di programmazione si può risparmiare sulla occupazione di memoria.
Potete notare come la riga 130 utilizzi una istruzione WAIT, invece della più familiare 130GETA$:IFA$ = "" THEN130.
Dovreste sperimentare questa istruzione prima di inserirla nei vostri programmi, tuttavia, poiché i risultati sono piuttosto imprevedibili. La locazione di memoria 197 contiene informazioni riguardo alla tastiera. Facendo uso di una istruzione WAIT197,64 il programma si arresterà finché non verrà premuto un tasto qualsiasi, incluso RETURN, che potrebbe essere ancora premuto dopo le istruzioni di INPUT delle righe 60 e 80. Questo problema si può risolvere utilizzando un valore numerico 32 invece di 64. In questo modo, circa metà dei tasti provocheranno la continuazione del programma, inclusa la barra-spazio. Il tasto RETURN sarà invece inefficace.
L'uso di altri numeri può rendere inefficienti altri tasti o addirittura impedire del tutto l'arresto del programma.
Avrete probabilmente notato che alcune istruzioni PRINT non hanno gli apici di chiusura. Essi non sono necessari sul VIC e sul 64 al termine di una riga, se non vengono usati per delimitare la lunghezza di una serie di spazi bianchi. È anche superfluo, parecchie volte, il comando di REVERSE OFF. Se il comando PRINT non è seguito da un punto e virgola, REVERSE viene automaticamente disattivato (OFF) al termine dell'istruzione.
La riga 769 è un commento (REMARK), ma senza istruzione **REM**. Non è necessaria tale istruzione poiché il programma non raggiunge mai questa riga, e quindi non rileva mai l'errore.
Le righe seguenti quest'ultima sono sottoprogrammi, e la riga 760 rimanda il programma ad una riga al di sopra di queste. Anche queste tecniche possono consentire un risparmio di memoria.

Re-Beep versione per VIC 20

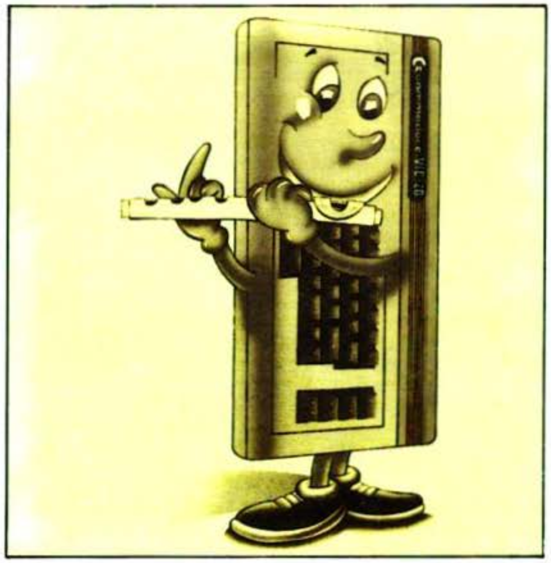
```
20 A=240:B=225:C=195:D=135:S=36875:R=RND(
   -TI):POKE36878,9:POKE36879,10:W=1
                                   :rem  248
30 PRINT"{CLR}"                    :rem  199
40 FORL=3TO7STEP2:POKE646,L:PRINTSPC(L+1)
   "{ 4 GIU'}{RVS}RE-BEEP{OFF}":NEXT:FORM
   =1TO3000:NEXT                   :rem  182
50 PRINT"{CLR}{WHT}{ 3 GIU'}LIVELLO DI DI
   FFICOLTA' (1-3)"                :rem  131
55 INPUTLV:IFLV<1ORLV>3THEN50      :rem  188
60 INPUT"{CLR}{ 3 GIU'}1 O 2 GIOCATORI";P
                                   :rem  222
70 IFP<>1ANDP<>2THEN60             :rem   85
80 IFP=2THENINPUT"{GIU'}NOME 1[<V>] GIOCA
   TORE";N1$:INPUT"{GIU'}NOME 2[<V>] GIOC
   ATORE";N2$                      :rem   29
```

```
89 REM TRONCA NOMI ECCESSIVAMENTE LUNGHI
                                   :rem  37
90 IFLEN(N1$)>12THENN1$=LEFT$(N1$,12)
                                   :rem  77
100 IFLEN(N2$)>12THENN2$=LEFT$(N2$,12)
                                   :rem 120
110 PRINT"{CLR}{CYN}{DES}{RVS}***R E - B
    E E P***{ 2 GIU'}"             :rem 194
120 PRINT"{RVS}{GRN}PREMI LA BARRA SPAZIA
    -TRICE PER INIZIARE"           :rem 180
130 WAIT197,32                     :rem 251
140 PRINT"{ 2 SU}{ 41 SPAZI}";:POKE36879,
    107                            :rem   1
149 REM COSTRUZIONE DEL CAMPO DI GIOCO
                                   :rem  42
150 PRINT"{BLK}":FORL=1TO7STEP2:PRINT"F"L
    "{ 3 GIU'}":NEXT               :rem 167
160 PRINT"{HOME}{ 4 GIU'}{GRN}":FORL=1TO4
    :PRINT"{ 4 DES}{RVS}{ 2 SPAZI}
    { 3 GIU'}":NEXT                :rem 204
170 PRINT"{ 2 SU}{ 9 DES}PER RIPARTIRE
                                   :rem   7
180 PRINT"{ 9 DES}PREMI '*'"       :rem 100
189 REM STAMPA NOMI E NUMERI PER I DUE GI
    OCATORI                        :rem  67
190 IFP=2THENPRINT"{HOME}{ 2 GIU'}
    { 9 DES}{RVS}{CYN}"N1$:PRINT"
    { 9 GIU'}{ 9 DES}{RVS}"N2$     :rem 213
200 IFP=2THENPOKE7788,177:POKE8008,178
                                   :rem 236
209 REM ILLUMINA IL NUMERO CORRISPONDENTE
     AD UN OPPORTUNO GIOCATORE     :rem  15
210 IFW=1THENPOKE38728,6:POKE38508,1
                                   :rem 129
220 IFW=-1THENPOKE38728,1:POKE38508,6
                                   :rem 175
229 REM COSTITUZIONE DELLA STRINGA DI SUO
    NI CASUALE                     :rem 196
230 R=INT(RND(1)*4)+1              :rem 128
240 S$=S$+STR$(R)                  :rem 172
250 FORM=1TO750/LV↑2:NEXT          :rem  82
259 REM VIC SUONA LA SUA STRINGA MUSICALE
                                   :rem 205
260 FORL=2TOLEN(S$)STEP2           :rem 242
270 Q=VAL(MID$(S$,L,1))            :rem 246
280 GETZ$:REM VENGONO ELIMINATI I CARATTE
    RI SPURII                      :rem 146
290 ONQGOSUB480,540,600,660:FORM=1TO75/LV
    ↑2:NEXT                        :rem 185
300 NEXT                           :rem 210
309 REM IL GIOCATORE COMPIE UN TENTATIVO
                                   :rem 208
310 GETZ$:IFZ$=""THEN310           :rem 125
320 IFZ$="{F1}"THENGOSUB480:F$=F$+STR$(1)
                                   :rem   8
330 IFZ$="{F3}"THENGOSUB540:F$=F$+STR$(2)
                                   :rem   8
340 IFZ$="{F5}"THENGOSUB600:F$=F$+STR$(3)
                                   :rem   8
350 IFZ$="{F7}"THENGOSUB660:F$=F$+STR$(4)
                                   :rem  17
360 IFZ$="*"THENRUN                :rem 117
369 REM CONFRONTO TRA LA STRINGA MUSICALE
     VIC E QUELLA DEL GIOCATORE :rem 122
370 IFLEN(F$)<>LEN(S$)THEN310      :rem 167
380 IFF$<>S$ANDE=0THENF$="":E=1:PRINT"
    {HOME}{ 9 GIU'}{ 9 DES}{YEL}{RVS}RIPR
    OVA":GOSUB770:GOTO250          :rem   2
390 PRINT"{HOME}{ 9 GIU'}{ 9 DES}
    { 9 SPAZI}"                    :rem  30
399 REM IL SECONDO TENTATIVO E' FALLITO
                                   :rem  94
400 IFF$<>S$ANDE=1THEN720          :rem 204
409 REM PUNTEGGIO (W=-1 INDICA IL SECONDO
     GIOCATORE)                    :rem  91
410 E=0:F$="":IFW=1THENS1=S1+1:IFH1<S1THE
    NH1=S1                         :rem  53
420 IFW=-1THENS2=S2+1:IFH2<S2THENH2=S2
                                   :rem  88
430 PRINT"{HOME}";:IFW=-1THENPRINT"
    { 10 GIU'}";                   :rem  86
440 IFW=1THENPRINT"{ 4 GIU'}{ 9 DES}{PUR}
    {RVS}RECORD ="H1:PRINT"{GIU'}
    { 9 DES}{GRN}PUNTI ="S1        :rem 232
450 IFW=-1THENPRINT"{ 4 GIU'}{ 9 DES}
    {PUR}{RVS}RECORD ="H2:PRINT"{GIU'}
    { 9 DES}{GRN}PUNTI ="S2        :rem  24
460 FORM=1TO750/LV↑2:NEXT          :rem  85
470 GOTO210                        :rem 103
479 REM SOTTOPROGRAMMA DEGLI EFFETTI SONO
    RI                             :rem  13
480 POKE38514,1:POKE38515,1        :rem 249
490 POKES,A                        :rem 140
500 GOSUB780                       :rem 180
510 POKES,0                        :rem 116
520 POKE38514,5:POKE38515,5        :rem 252
530 RETURN                         :rem 120
540 POKE38602,1:POKE38603,1        :rem 242
550 POKES,B                        :rem 138
560 GOSUB780                       :rem 186
570 POKES,0                        :rem 122
580 POKE38602,5:POKE38603,5        :rem 254
590 RETURN                         :rem 126
600 POKE38690,1:POKE38691,1        :rem 253
610 POKES,C                        :rem 136
620 GOSUB780                       :rem 183
630 POKES,0                        :rem 119
640 POKE38690,5:POKE38691,5        :rem   9
650 RETURN                         :rem 123
660 POKE38778,1:POKE38779,1        :rem  17
670 POKES,D                        :rem 143
680 GOSUB780                       :rem 189
690 POKES,0                        :rem 125
700 POKE38778,5:POKE38779,5        :rem  20
710 RETURN                         :rem 120
720 POKE36874,159:FORM=1TO999:NEXT:POKE36
    874,135:FORM=1TO1500:NEXT:POKE36874,0
                                   :rem  11
730 S1=0:S2=0:S$="":F$="":E=0:REM RESET
                                   :rem 155
740 PRINT"{HOME}";:IFW=-1THENPRINT"
    { 10 GIU'}";                   :rem  90
750 PRINT"{ 6 GIU'}{ 9 DES}{ 10 SPAZI}":I
    FP=2THENW=-W:REM ALTERNA I GIOCATORI
                                   :rem 182
760 GOTO210                        :rem 105
769 SOTTOPROGRAMMA DI EFFETTI SONORI E TE
    MPORIZZAZIONE                  :rem  50
770 POKE36874,135:FORM=1TO250:NEXT:POKE36
    874,0:RETURN                   :rem  24
780 FORM=1TO600/LV↑2:NEXT:RETURN:rem 110
```

Re-Beep versione per C 64

```
100 REM RE-BEEP  64                :rem 191
110 S=54272:R=RND(-RND(0)):FORI=0TO4:READ
    H(I),L(I):NEXT:S$=""           :rem  89
120 FORI=0TO21:POKES+I,0:NEXT      :rem  59
130 POKES+24,15:POKE53280,2:POKE53281,0:W
    =1:D$="{HOME}{ 25 GIU'}"       :rem 168
140 PRINT"{CLR}"                   :rem 249
```

```
150 FORL=3TO7STEP2:POKE646,L:PRINTSPC(L*2
    +5)"{ 4 GIU'}{RVS}{ 2 SPAZI}RE-BEEP "
    :NEXT                        :rem  251
160 FORM=1TO1500:GETA$:IFA$=""THENNEXT
                                 :rem   62
170 INPUT"{CLR}{WHT}{ 3 GIU'} LIVELLO DI
    DIFFICOLTA' (1-3)";LV:IFLV<1ORLV>3THE
    N170                         :rem  233
180 INPUT"{CLR}{ 3 GIU'}1 O 2 GIOCATORI";
    P:IFP<>1ANDP<>2THEN180       :rem  108
190 IFP=2THENINPUT"{GIU'} NOME 1' GIOCATO
    RE";N1$:INPUT"{GIU'} NOME 2' GIOCATOR
    E";N2$                       :rem   33
200 N1$=LEFT$(N1$,12):N2$=LEFT$(N2$,12)
                                 :rem   50
210 PRINT"{CLR}"SPC(10)"{CYN}{RVS}
    { 4 SPAZI}R E ** B E E P{ 2 SPAZI}
                                 :rem  139
220 PRINT{ 2 GIU'}"{RVS}{GRN}PREMI LA BAR
    RA SPAZIATRICE PER INIZIARE":rem  136
230 WAIT197,32:POKE198,0         :rem   99
240 PRINT"{SU}{ 39 SPAZI}";:POKE53280,3:P
    OKE53281,6                   :rem  203
250 REM COSTRUZIONE DEL CAMPO DI GIOCO
                                 :rem   35
260 PRINT"{BLK}":FORL=1TO7STEP2:PRINTSPC(
    15)"F"L"{ 2 GIU'}":NEXT      :rem   53
270 PRINT"{HOME}{ 3 GIU'}{GRN}":FORL=1TO4
    :PRINTSPC(20)"{RVS}{ 3 SPAZI}
    { 2 GIU'}":NEXT              :rem  209
280 PRINTSPC(15)"{ 2 GIU'}{BLK}PER RIPART
    IRE                          :rem   49
290 PRINTSPC(15)"{GIU'}PREMI '*':rem  237
300 REM STAMPA NOMI E NUMERI PER I DUE GI
    OCATORI                      :rem   52
310 IFP=2THENPRINT"{HOME}{ 2 GIU'}{RVS}
    {CYN}"SPC(18)N1$:PRINT"{ 12 GIU'}
    {RVS}"SPC(18)N2$             :rem   56
320 IFP=2THENPOKE1120,177:POKE1640,178
                                 :rem  208
330 REM ILLUMINA IL NUMERO CORRISPONDENTE
     AD UN OPPORTUNO GIOCATORE   :rem   10
340 IFW=-1THENPOKE55392,6:POKE55912,1
                                 :rem  172
350 IFW=1THENPOKE55912,6:POKE55392,1
                                 :rem  128
360 REM COSTITUZIONE DELLA STRINGA DI SUO
    NI CASUALE                   :rem  192
370 R=INT(RND(1)*4)+1            :rem  133
380 S$=S$+STR$(R)                :rem  177
390 GOSUB760                     :rem  185
400 REM C64 SUONA LA SUA STRINGA MUSICALE
                                 :rem  140
410 FORL=2TOLEN(S$)STEP2         :rem  239
420 NT=VAL(MID$(S$,L,1))         :rem   68
430 FG=1:GOSUB630:FORA=1TO75/LV↑2:NEXT
                                 :rem  158
440 NEXT:POKE198,0               :rem   62
450 REM VENGONO ELIMINATI I CARATTERI SPU
    RII                          :rem  249
460 GETZ$:IFZ$=""THEN460         :rem  137
470 FORI=1TO4:IFZ$=MID$("{F1}{F3}{F5}{F7}
    ",I,1)THENNT=I:I=10          :rem  139
480 NEXT:IFI>10THENFG=1:GOSUB630:F$=F$+ST
    R$(NT):GOTO510               :rem  152
490 IFZ$="*"THENRUN              :rem  121
500 REM CONFRONTO TRA LA STRINGA MUSICALE
     VIC E QUELLA DEL GIOCATORE :rem  109
510 IFLEN(F$)<>LEN(S$)THEN460    :rem  169
520 IFF$=S$THEN570               :rem   15
530 IFE=1THEN690                 :rem  168
```



```
540 IFE=0THENF$="":E=1:GOSUB760:PRINT"
    {HOME}{GIU'}"SPC(15)"{YEL}{RVS}RIPROV
    A":GOSUB740                  :rem  112
550 PRINT"{HOME}{GIU'}"SPC(15)"
    { 9 SPAZI}":GOSUB750:POKE198,0:GOTO39
    0                            :rem   60
560 REM PUNTEGGIO (W=-1 INDICA IL SECONDO
     GIOCATORE)                  :rem   89
570 E=0:F$="":IFW=1THENS1=S1+1:IFH1<S1THE
    NH1=S1                       :rem   60
580 IFW=-1THENS2=S2+1:IFH2<S2THENH2=S2
                                 :rem   95
590 IFW=1THENPRINT"{HOME}{ 2 GIU'}{PUR}
    {RVS}RECORD{ 2 SPAZI}:"H1:PRINT"{GRN}
    PUNTI :"S1                   :rem  190
600 IFW=-1THENPRINTLEFT$(D$,16)"{PUR}
    {RVS}RECORD{ 2 SPAZI}:"H2:PRINT"{GRN}
    PUNTI :"S2                   :rem   75
610 GOSUB760:GOTO340             :rem  190
620 REM SOTTO PROGRAMMA DEGLI EFFETTI SON
    ORI                          :rem    1
630 WF=32:IFFGTHENPRINTLEFT$(D$,NT*3+2)SP
    C(20)"{RVS}{WHT}{ 3 SPAZI}":WF=16
                                 :rem  233
640 VC=VC+7:IFVC=21THENVC=0      :rem  162
650 POKES+VC+5,48:POKES+VC+6,170:POKES+VC
    ,L(NT):POKES+VC+1,H(NT)      :rem  172
660 POKES+VC+4,WF+1:GOSUB750:POKES+VC+4,W
    F                            :rem  100
670 IFFGTHENPRINTLEFT$(D$,NT*3+2)SPC(20)"
    {RVS}{GRN}{ 3 SPAZI}{CYN}"   :rem  177
680 RETURN                       :rem  126
690 FG=0:NT=0:GOSUB630:GOSUB760:NT=2:GOSU
    B630:GOSUB760                :rem  129
700 S1=0:S2=0:S$="":F$="":E=0    :rem  247
710 IFP=2THENW=-W:REM SWITCH PLAYERS
                                 :rem   61
720 GOTO340                      :rem  105
730 REM EFFETTI SONORI DI TIMING:rem  180
740 FG=0:NT=2:GOSUB630:RETURN    :rem   77
750 IFFGTHENFORM=1TO600/LV↑2:NEXT:RETURN
                                 :rem  182
760 FORM=1TO400:NEXT:RETURN      :rem    9
770 DATA 21,31, 14,24, 15,210, 16,195, 18
    ,209                         :rem  217
```

SOFTWARE edizioni JCE

GIOCHI



GARDEN WARS
Siete vittime di un incantesimo. Per liberarvene dovete sfuggire a mostri orrendi e attraversare otto giardini con innumerevoli insidie. Solo la vostra abilità vi potrà salvare. Specifico per COMMODORE 64.
cod J/3000-04 L. 20.000



CHESS WARS
Giocate a scacchi con il vostro Commodore VIC 20 inespanso. Con questo programma potrete cimentarvi con il computer scegliendo fino a nove livelli di difficoltà. Sono previste tutte le mosse speciali stabilite dal regolamento.
cod J/3000-03 L. 20.000

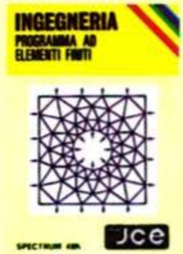


BATTAGLIA NAVALE
È possibile giocare alla Battaglia Navale con lo ZX Spectrum 48K. Il calcolatore segnala i tiri effettuati e i centri ottenuti.
cod J/3000-01 L. 15.000



MOSAICO
Scoprite, pezzo dopo pezzo, le numerosissime caselle che compongono il Mosaico. Il gioco è appassionante, elettrizzante ed altamente creativo. Specifico per SPECTRUM 48K
cod J/3000-02 L. 15.000

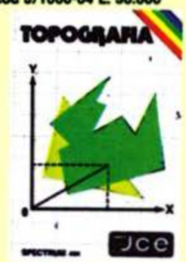
TECNICA



INGEGNERIA, programma ad elementi finiti.
Questo programma permette di analizzare in maniera precisa e rapida, pannelli piani di qualsiasi forma e caratteristiche, fornendo moltissimi dati riguardanti la geometria della struttura. Specifico per SPECTRUM 48K.
cod J/1000-04 L. 30.000



CALCOLO TRAVI IPE
Il programma consente il calcolo di una o più travi IPE, caricate uniformemente o semplicemente appoggiate agli estremi. Specifico per SPECTRUM 48K.
cod J/1000-02 L. 24.000



TOPOGRAFIA
Questo Package permette il calcolo dell'area di una figura piana e fornisce le coordinate cartesiane o polari dei vertici. Specifico per SPECTRUM 48K.
cod J/1000-01 L. 30.000



ENERGIA SOLARE
Programma che analizza nei minimi dettagli le varie componenti tecniche ed economiche di un impianto solare, stabilendo la convenienza della sua realizzazione. Specifico per SPECTRUM 48K.
cod J/1000-03 L. 30.000

HOME UTILITY



SPECTRUM WRITER
È un programma professionale di elaborazione e stampa delle parole e dei testi di lettere, articoli, saggi e documenti (Word Processor). Specifico per SPECTRUM 48K.
cod J/2000-02 L. 40.000



TOTIP
Un programma per giocare la schedina TOTIP, una colonna per volta oppure dei sistemi con triple o doppie. La schedina è ragionata, ma propone anche qualche risultato a "sorpresa". Specifico per SPECTRUM 48K.
cod J/2000-01 L. 17.000



GRAFICA PER TUTTI
Con questo programma il vostro Spectrum 48K diventa un validissimo strumento per disegnare, e il vostro schermo una tela sulla quale dipingere con i colori dell'arcobaleno.
cod J/2000-03 L. 20.000

CON ISTRUZIONI IN ITALIANO.

Cedola di commissione da inviare a:
JCE - Via dei Lavoratori, 124 - 20092 Cinisello B. - MI
Inviatemi i seguenti programmi:

Cod. progr.	Q.tà	Cod. progr.	Q.tà	Cod. progr.	Q.tà

Desidero ricevere il materiale indicato nella tabella, a mezzo pacco postale contro assegno, al seguente indirizzo:

Nome
Cognome
Via
Città
Data C.A.P.

Desidero ricevere la fattura si no

Partita I.V.A. o per i privati Codice Fiscale

PAGAMENTO:
A) Anticipato, mediante assegno circolare o vaglia postale per l'importo totale dell'ordinazione
B) Contro assegno al postino l'importo totale dell'ordinazione
AGGIUNGERE L. 2.000 per contributo fisso. I prezzi sono comprensivi di I.V.A.

JCE
Via dei Lavoratori, 124
20092 Cinisello Balsamo - MI

React

di Don Whitaker
trad. e adatt. E. Comini

"React" è un gioco d'azione e di strategia per il VIC 20, nella sua configurazione base, è richiesto l'uso del joystick

React è un coloratissimo gioco d'azione nel quale, controllando col vostro joystick un individuo chiamato "VIC-Y", dovete ripulire lo schermo da tutti gli oggetti presenti su di esso. Ciò che rende "React" un gioco difficile è il dover schivare il recinto elettrificato, le guardie (che osservano con facce sorridenti, ma che attualmente vi guardano di traverso), e la traccia colorata che muovendovi lasciate dietro di voi.

Come si gioca

Il gioco comincia con 5 VIC-Y. Per partire si preme il pulsante del joystick. Una volta apparso lo schermo di gioco usate il joystick per muovere VIC-Y da un oggetto all'altro. Dopo aver compiuto pochi passi scoprirete che "React" non è solo un gioco d'azione ma anche di strategia. Se voi raccogliete tutti gli oggetti disposti sul video muovendovi a caso potrete notare che vi chiuderete in una gabbia che vi impedirà di procedere per raccogliere gli oggetti rimanenti. Se finite contro il recinto elettrificato oppure contro le guardie, o, ancora, contro la vostra traccia, VIC-Y è eliminato.
Sullo schermo vengono mostrati i punti, il numero di VIC-Y rimanenti ed i punti di "bonus". Ad ogni schermo aumenta la difficoltà perché alla gabbia elettrica si aggiungono delle palizzate, inoltre il numero di guardie aumenta. L'unico sistema per fare molti punti o giocare competitivamente è quello di collezionare i punti del "bonus" che vengono assegnati se lo schermo è ripulito in meno di 60 secondi; sullo schermo non appare l'orologio, ma potete ascoltarne il ticchettio mentre giocate. Più velocemente finite, più alto è il "bonus" ottenuto.

Utilizzo dei programmi

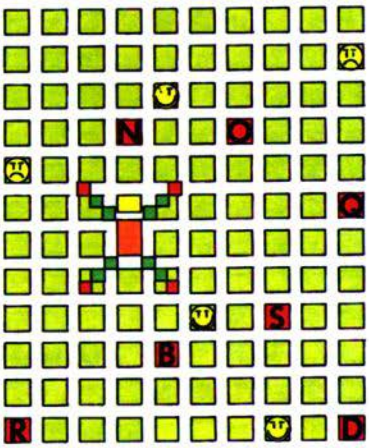
Battete il 1° programma e fate il SAVE 2 volte (tanto per essere sicuri). A questo punto date il NEW, battete il 2° programma, e salvatelo 2 volte.
Successivamente caricate con LOAD il 1° programma e date il RUN. Sullo

Linee	Commento
1° programma (SET)	
10-40	Pulizia delle schermo, cambio dei colori; stampa delle opzioni per centrare il quadro.
50	Pulizia schermo e colorazione in nero.
70	Riserva un'area di memoria per il nuovo SET di caratteri.
80	Copiatura dei caratteri (lettere maiuscole e simboli non grafici) nelle locazioni di memoria da 7168 a 7679.
120-166	Definizione dei nuovi caratteri.
170	Cancellazione del 1° programma e istruzione per il LOAD ed il RUN del 2° programma.
10000-10009	Sottoprogramma che provvede alla centratura del quadro.
2° Programma (MAIN)	
5	Scelta del massimo volume per i suoni del gioco.
10	Pulizia schermo e definizione dei colori.
20-28	Stampa del titolo.
30-40	Effetto sonoro e cambio dei colori del titolo.
109-130	Inizializza le variabili.
135-136	Generazione di un ritardo mentre attende che venga premuto il pulsante del joystick.
160	Cambiamento del colore dello schermo in nero e cambiamento del set di caratteri (standard/definiti dal programma).
180	Salto al sottoprogramma che disegna lo schermo.
190-210	Scelta casuale dei colori e disposizione sullo schermo delle guardie.
220-240	Scelta casuale dei colori e disposizione sullo schermo degli oggetti.
249	Posiziona a zero l'orologio in tempo reale.
250	Inizio del ciclo principale. Sistemazione del VIC-Y nella posizione di partenza e lettura del joystick.
260	Controllo collisione con la gabbia, le guardie o la traccia. Se si è verificata una collisione salto al sottoprogramma di "perdita di un VIC-Y".
270	Controllo per vedere se il VIC-Y si è mosso.
280	Controllo per vedere se il VIC-Y ha raggiunto un oggetto.
290	Generazione del suono di movimento, incremento del punteggio, aggiornamento della posizione di VIC-Y e ritorno alla linea 250.
390	Controllo per vedere se è stato premuto il pulsante del joystick.
400-420	Controllo della posizione del joystick.
440-450	Esecuzione del disegno dello schermo base del gioco.
460-550	Aggiunta di più palizzate alla gabbia elettrificata.
560-580	Sottoprogramma "perdita di un VIC-Y". Cambiamento del bordo dello schermo in giallo. Rumore d'esplosione. Controllo per vedere se sono finiti i VIC-Y.
600-610	Sottoprogramma che viene eseguito quando si sono raccolti tutti gli oggetti. Causa: cambiamenti di colore genera suoni, determina il "bonus" e lo aggiunge ai punti.

schermo apparirà una richiesta: "Volete centrare il quadro (S/N)". Se la vostra TV ha il quadro non ben centrato allora centratelo utilizzando i tasti CRSR per sistemare il quadro. Premendo RETURN verrà automaticamente caricato e lanciato il 2° programma e voi potrete cominciare a giocare a "REACT". Se utilizzate i dischi, dopo aver "lanciato" il primo programma premete RUN-STOP e caricate il 2° programma.

Come funziona React?

Il primo programma crea il SET di caratteri e vi permette di centrare il quadro. Il secondo programma è il MAIN. Segue una descrizione dei due programmi:



620-650	Determina la visualizzazione, sullo schermo appena concluso, del punteggio, del numero di VIC-Y rimanenti e del "bonus".
660-670	Attesa fino alla pressione del pulsante sul joystick.
680-720	Sottoprogramma di fine gioco, vengono mostrati i punti corretti ed il miglior punteggio. Attesa finché non viene premuto il bottone di fuoco per iniziare un nuovo gioco.

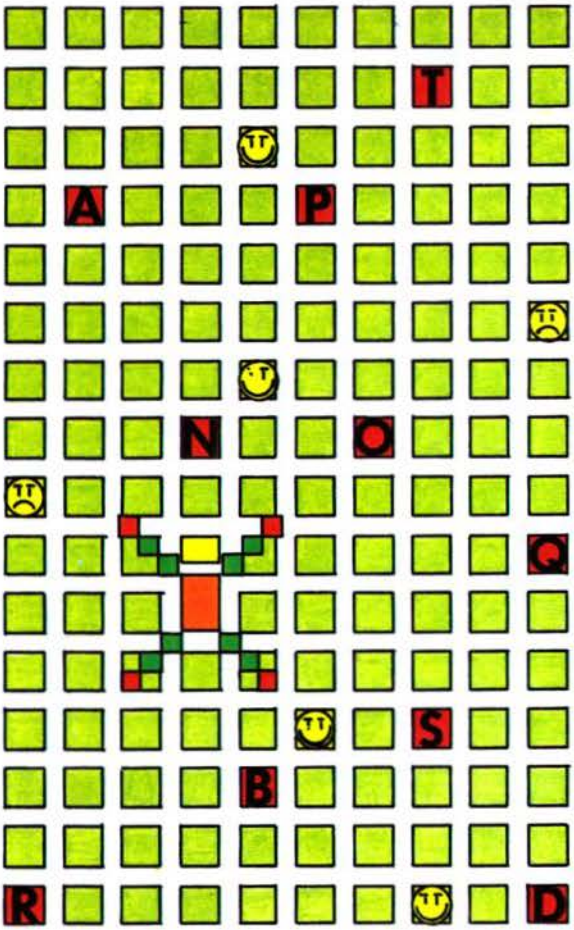
Variabili utilizzate

CO	Locazione di memoria 36879 utilizzata per i colori dello sfondo e del bordo dello schermo.
SM	Valore del codice dello schermo per il disegno di una guardia (28).
ML	Numero di VIC-Y rimanenti.
C	Differenza tra la memoria di schermo e la memoria per il colore.
ER	Valore del codice dello schermo per il disegno di uno spazio (32).
MA	Valore del codice dello schermo per il disegno di un VIC-Y.
TR	Valore del codice dello schermo per il disegno di una traccia del VIC-Y.
WA	Valore del codice dello schermo per il disegno della gabbia elettrificata.
S	Locazione di memoria della seconda voce del generatore di suono (36875).
V	Locazione di memoria per il controllo del volume (36878).
NE	Costante per l'aggiunta di punti.
SC	Punteggio corrente.
NM	Numero di guardie.
NC	Numero di oggetti.
CL	Numero corrente degli oggetti ancora presenti sullo schermo.
JS (X,X)	Matrice usata per trasformare la posizione del joystick nel numero che corrisponde al movimento del VIC-Y.
DD, PA, PB, OP,TF, N1, N2, N3, ed N4	sono costanti utilizzate per la lettura del joystick.

React versione per VIC 20 Set

```
10 PRINT"{CLR}{WHT}":POKE36879,107
                                :rem    8
20 PRINT"{ 3 SPAZI}VOLETE CENTRARE{HOME}
   { 3 GIU'}{ 3 DES}IL QUADRO (S/N)"
                                :rem  117
30 GETA$:IFA$<>"S"ANDA$<>"N"THEN30
                                :rem  189
40 IFA$="S"THENGOSUB10000       :rem  204
50 PRINT"{CLR}":POKE36879,8     :rem  167
60 PRINT"CARICAMENTO CARATTERI ATTENDERE
   PREGO"                       :rem   21
70 POKE52,28:POKE56,28:CLR      :rem   24
80 FORI=7168TO7679:POKEI,PEEK(I+25600):NE
   XT                           :rem  105
120 FORI=7168TO7175:READA:POKEI,A:NEXT:DA
    TA24,90,36,24,24,24,36,66   :rem   67
130 FORI=7384TO7551:READA:POKEI,A:NEXT
                                :rem  123
140 DATA0,0,24,24,60,90,60,66   :rem  176
141 DATA60,66,165,129,165,154,66,60
                                :rem  246
142 DATA255,129,165,129,189,165,189,255
                                :rem  209
143 DATA0,126,126,126,126,126,126,0
                                :rem  220
144 DATA170,255,85,255,170,255,85,255:DAT
    A0,0,0,0,0,0,0,0            :rem  105
145 DATA0,6,8,60,126,126,126,60 :rem   29
146 DATA24,60,126,255,255,255,60,60
                                :rem  238
147 DATA56,60,254,255,255,254,60,56
                                :rem  250
148 DATA60,60,255,255,255,126,60,24
                                :rem  240
149 DATA28,60,127,255,255,127,60,28
                                :rem  248
150 DATA0,0,56,40,8,20,34,127   :rem  174
151 DATA28,34,42,34,28,20,119,0 :rem   26
152 DATA219,195,36,153,153,36,195,219
                                :rem   96
153 DATA75,75,75,75,180,180,180,180
                                :rem  251
154 DATA126,102,66,126,24,36,195,66
                                :rem  243
155 DATA60,66,189,255,255,189,66,60
                                :rem    9
156 DATA0,0,56,68,146,68,56,0   :rem  199
157 DATA0,60,60,60,60,60,255,0  :rem  229
158 DATA129,195,231,255,255,231,195,129
                                :rem  198
159 DATA0,102,102,0,66,102,60,24:rem   62
160 FORI=7632TO7679:READA:POKEI,A:NEXT
                                :rem  133
161 DATA171,171,173,173,181,181,213,213
                                :rem  174
162 DATA195,195,0,24,24,0,195,195
                                :rem  143
```

```
163 DATA255,145,145,159,249,137,137,255
                                   :rem  200
164 DATA0,0,151,146,242,146,151,0
                                   :rem  117
165 DATA165,36,231,24,24,231,36,165
                                   :rem  236
166 DATA195,231,126,60,60,126,231,195
                                   :rem   83
170 POKE198,5:POKE631,78:POKE632,69:POKE6
    33,87:POKE634,13:POKE635,131:END
                                   :rem   22
10000 PRINT"{CLR}CENTRATE IL QUADRO CON
      {GIU'} I TASTI CRSR{ 2 SPAZI}QUANDO
       {GIU'}FINITE PREMETE{ 2 SPAZI}RETU
      RN "                         :rem  143
10001 A=PEEK(197):B=PEEK(653):IFA=15THEN1
      0009                         :rem   42
10002 IFA=31THENA=36865:GOTO10005
                                   :rem  134
10003 IFA=23THENA=36864:GOTO10005
                                   :rem  135
10004 GOTO10001                    :rem   32
10005 IFB=1THENB=-1                :rem   65
10006 IFB=0THENB=1                 :rem   20
10007 Q=PEEK(A):IFQ+B<0OR(Q+B>17ANDA=3686
      4)THEN10001                  :rem    3
10008 POKEA,Q+B:GOTO10001          :rem  184
10009 PRINT"{CLR}":RETURN          :rem  120
```



LOAD
React

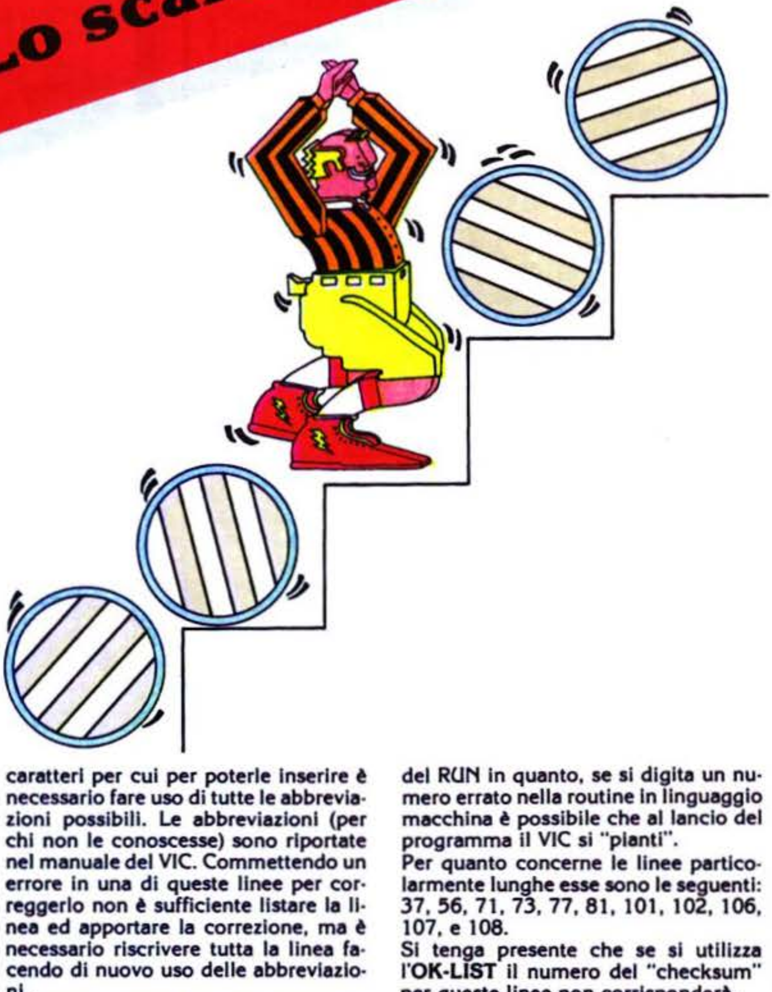
Main

```
5 POKE36878,15                     :rem    6
10 POKE36879,238:PRINTCHRS(147)CHR$(144)
                                   :rem   10
20 PRINT"RRRRR"                    :rem  205
21 FORI=1TO3:PRINT"R{ 3 SPAZI}R":NEXT
                                   :rem  255
24 PRINT"RRRRR E A C T"            :rem  238
25 PRINT"RR"                       :rem  220
26 PRINT"R R"                      :rem  221
27 PRINT"R{ 2 SPAZI}R"             :rem  222
28 PRINT"R{ 3 SPAZI}R"             :rem  223
30 FORI=8TO248STEP16:FORJ=0TO7:POKE36879,
   I+J                             :rem    2
40 POKE36875,160+J+(I/16):NEXT:NEXT:POKE3
   6875,0:POKE36879,62             :rem    5
50 PRINT:PRINT"PREMERE IL BOTTONE DI FUOC
   O PER INIZIARE"                 :rem    8
109 CO=36879:A=7901:SM=28:ML=5     :rem  201
110 C=30720:ER=32:MA=0:TR=30:CH=33:WA=31:
    S=36875:V=36878:DD=37154:PA=37137
                                   :rem  213
120 PB=37152:OP=127:TF=255:N1=128:N2=8:N3
    =16:N4=4:NE=1:SC=0:DIMJS(2,2):NM=2:NC
    =1:CL=NC                       :rem   43
130 FORI=0TO2:FORJ=0TO2:READJS(J,I):NEXT:
    NEXT:DATA-23,-22,-21,-1,0,1,21,22,23
                                   :rem  250
135 GOSUB390:IFFR=0THEN135         :rem   75
136 POKES,230:FORI=1TO4:NEXT:POKES,0
                                   :rem   29
160 PRINTCHR$(147):POKECO-10,255:POKECO,8
                                   :rem  194
180 GOSUB430                       :rem  176
190 A=7703:FORI=1TONM              :rem  255
200 W=INT(RND(I)*505)+1:J=INT(RND(I)*6)+2
    :IFPEEK(7680+W)<>32THEN200     :rem  110
205 IFW+7680=ATHEN200              :rem  188
210 POKES,150:POKES+1,230:POKEW+7680,28:P
    OKEW+C+7680,J:POKES,0:POKES+1,0:NEXT
                                   :rem  175
220 FORI=1TONC                     :rem  102
230 W=INT(RND(I)*505)+1:J=INT(RND(I)*6)+2
    :K=INT(RND(I)*64)+1:IFPEEK(7680+W)<>3
    2THEN230                       :rem  224
235 IFK>26ANDK<33ORK=0ORW+7680=ATHEN230
                                   :rem  109
240 POKES,240:POKES+1,200:POKEW+7680,K:PO
    KEW+C+7680,J:POKES,0:POKES+1,0:NEXT
                                   :rem  144
249 TI$="000000"                   :rem    1
250 POKEA,MA:GOSUB400:B=A+JS(X+1,Y+1)
                                   :rem  183
260 IFPEEK(B)=WAORPEEK(B)=TRORPEEK(B)=SMT
    HEN560                         :rem  240
270 IFPEEK(B)=MATHEN290            :rem  117
280 IFPEEK(B)<>ERTHENPOKES,240:SC=SC+50:C
    L=CL-NE:POKES,0:IFCL=.THEN600
                                   :rem  141
290 POKES,200:POKES,0:SC=SC+NE:POKEA,TR:A
    =B:GOTO250                     :rem  212
390 P=PEEK(PA):FR=-((PAND32)=0):RETURN
                                   :rem   29
```

```
400 POKEDD,OP:S3=-((PEEK(PB)ANDN1)=MA):PO
    KEDD,TF                            :rem 190
410 P=PEEK(PA):S1=-((PANDN2)=MA):S2=((PAN
    DN3)=MA):S0=((PANDN4)=MA)          :rem 122
420 X=S2+S3:Y=S0+S1:RETURN             :rem 125
430 TC=INT(RND(I)*6)+2:FORI=7680TO8185:PO
    KEI+C,TC:NEXT                      :rem 51
440 FORI=7680TO7701:POKEI,WA:POKEI+C,NE:P
    OKEI+485,WA:POKEI+C+485,NE:NEXT
                                       :rem 7
450 FORI=7702TO8164STEP22:POKEI,WA:POKEI+
    C,NE:POKEI+21,WA:POKEI+C+21,NE:NEXT
                                       :rem 43
460 W=INT(RND(I)*5)+1                  :rem 163
470 ONWGOSUB490,510,540                :rem 51
480 RETURN                             :rem 124
490 FORI=7795TO8095STEP22:POKEI+C,NE:POKE
    I,WA:POKEI+C+11,NE:POKEI+11,WA:NEXT
                                       :rem 60
500 RETURN                             :rem 117
510 FORI=7690TO7822STEP22:POKEI+C,NE:POKE
    I,WA:POKEI+C+NE,NE:POKEI+NE,WA
                                       :rem 21
520 POKEI+C+330,NE:POKEI+330,WA:POKEI+C+3
    31,NE:POKEI+331,WA:NEXT            :rem 134
530 FORI=7923TO7928:POKEI+C,NE:POKEI,WA:P
    OKEI+C+14,NE:POKEI+14,WA:NEXT:RETURN
                                       :rem 180
540 FORI=7712TO7822STEP22:POKEI+C,NE:POKE
    I,WA:POKEI+C+NE,NE:POKEI+NE,WA:NEXT
                                       :rem 140
550 FORI=7840TO8148STEP22:POKEI+C,NE:POKE
    I,WA:POKEI+C+9,NE:POKEI+9,WA:NEXT:RET
    URN                                :rem 247
560 POKECO,15:IFPEEK(B)=28THENPOKEB,29
                                       :rem 77
570 POKEA,27:FORI=250TO115STEP-.3:POKES+2
    ,I:NEXT:POKES+2,0                  :rem 95
580 ML=ML-1:IFML=.THEN680              :rem 4
590 GOTO620                            :rem 111
600 POKEA,TR:POKEB,MA:FORI=1TO90:POKECO,I
    :POKES,150+I:POKES+1,150+I:POKES-1,15
    0+I                                :rem 33
601 NEXT:POKES,0:POKES+1,0:POKES-1,0:POKE
    CO,152                             :rem 151
610 BO=100-VAL(TI$):IFBO<0THENBO=0
                                       :rem 11
620 PRINTCHR$(147):POKECO,104:PRINTCHR$(5
    ):SC=SC+BO                         :rem 124
630 PRINT"PUNTI ";SC:PRINT             :rem 146
640 PRINT"VIC-Y RIMANENTI";ML:PRINT:PRINT
    "BONUS ";BO                        :rem 115
650 PRINT:PRINT:PRINT:PRINTCHR$(15)"PREME
    RE IL BOTTONE DI FUOCO PER CONTINUARE
    "                                  :rem 33
660 GOSUB390:IFFR=0THEN660             :rem 81
670 POKES,230:PRINTCHR$(147):POKECO,8:NM=
    NM+2:NC=NC+2:CL=NC:POKES,0:BO=0:GOTO1
    80                                 :rem 231
680 PRINTCHR$(147)CHR$(144):POKECO,30
                                       :rem 142
690 PRINT"PUNTI ";SC:IFSC>HSTHENHS=SC
                                       :rem 166
700 PRINT:PRINT"+{ 2 SPAZI}RECORD
    { 3 SPAZI}";HS:PRINT:PRINT:PRINT"PREM
    ERE IL{ 2 SPAZI}BOTTONE DI{ 3 SPAZI}
    FUOCO PER GIOCARE { 8 DES}ANCORA"
                                       :rem 206
710 GOSUB390:IFFR=0THEN710             :rem 73
720 POKES,200:BO=0:SC=0:NC=1:NM=2:CL=NC:P
    RINTCHR$(147):ML=5:POKECO,8:POKES,0:G
    OTO180                             :rem 223
```

Lo scalatore

di Chris Lesher
trad. e adatt. E. Comini



"Lo scalatore" è uno dei migliori giochi che siano stati realizzati per il VIC 20 non espanso e per C64 ed è un ottimo esempio di ciò che si può fare con il BASIC.

Voi vi trovate alla base di 4 livelli di travi collegate mediante scale. In cima c'è una pila di 12 barili e disseminate sulle travi si trovano alcune cassette di attrezzi.
L'obiettivo del gioco è quello di camminare sulle travi e di raccogliere tutte le cassette cercando di evitare i barili che rotolano. Riuscendo a raccogliere tutte le cassette si ottiene un "Bonus" ed il gioco riprende con uno schermo più complesso.
Il gioco è stato scritto interamente in BASIC con solo una piccola routine in linguaggio macchina per la lettura del joystick. Usando il joystick si può muovere lo scalatore in alto, in basso, a destra e a sinistra guidandolo lungo le travi e sulle scale. Premendo il pulsante lo scalatore fa un salto nella direzione dell'ultimo movimento che è stato effettuato.
Costui può saltare sia i barili che i buchi presenti nelle travi.

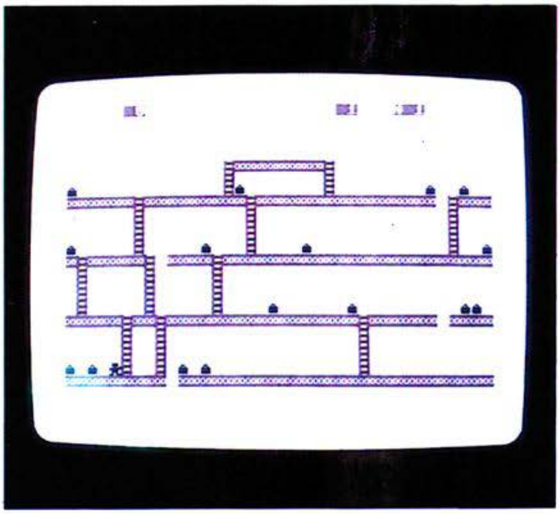
Punteggio

Si ottengono 150 punti per ogni cassetta di arnesi che viene raccolta, 1000 punti saltando un barile, e 100 punti per ogni barile rimasto dopo che sono state raccolte tutte le cassette di attrezzi. Il punteggio è visibile nell'angolo a sinistra dello schermo. Il numero dello schermo al quale si è giunti è visibile nell'angolo destro. Il numero di scalatori ancora a disposizione è visibile tra il punteggio ed il numero dello schermo. Il gioco inizia con 3 scalatori e se ne ottiene uno extra ogni 10.000 punti. Si perde uno scalatore quando lo stesso viene investito da un barile, quando cade da una trave oppure quando i barili sono rotolati tutti senza aver raccolto tutte le cassette con gli arnesi.
Il gioco termina quando si perde l'ultimo scalatore.
Molte linee BASIC sono più lunghe del limite massimo consentito pari a 88 caratteri per cui per poterle inserire è necessario fare uso di tutte le abbreviazioni possibili. Le abbreviazioni (per chi non le conoscesse) sono riportate nel manuale del VIC. Commettendo un errore in una di queste linee per correggerlo non è sufficiente listare la linea ed apportare la correzione, ma è necessario riscrivere tutta la linea facendo di nuovo uso delle abbreviazioni.
Per essere sicuri di non perdere il programma è opportuno salvarlo prima del RUN in quanto, se si digita un numero errato nella routine in linguaggio macchina è possibile che al lancio del programma il VIC si "pianti".
Per quanto concerne le linee particolarmente lunghe esse sono le seguenti: 37, 56, 71, 73, 77, 81, 101, 102, 106, 107, e 108.
Si tenga presente che se si utilizza l'**OK-LIST** il numero del "checksum" per queste linee non corrisponderà.

Lo scalatore versione per C 64

```
1 PRINT"{CLR}{ 7 GIU'}{ 12 DES}HARDHAT CL
  IMBER":POKE53280,1:POKE53281,1:GOSUB780
                                 :rem   76
2 POKE52,48:POKE56,48:CLR        :rem  231
5 PRINT"{CLR}"                   :rem  153
6 PRINT"{ 7 GIU'}{ 12 DES}HARDHAT CLIMBER
  "                              :rem  212
10 POKE53280,1:POKE53281,1-      :rem  184
20 DO(0)=-1:DO(1)=1:DI=DO(INT(RND(1)*2))
                                 :rem  180
30 A$=">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>{SIN}{INST}>":DIMB(11):G=54272:E2
   =0                            :rem   19
40 SC=1:CH=2:E1=0:D(0)=4:D(1)=2:D(4)=7:Z=
   57:E3=1:Q=10000:J=56          :rem   81
```

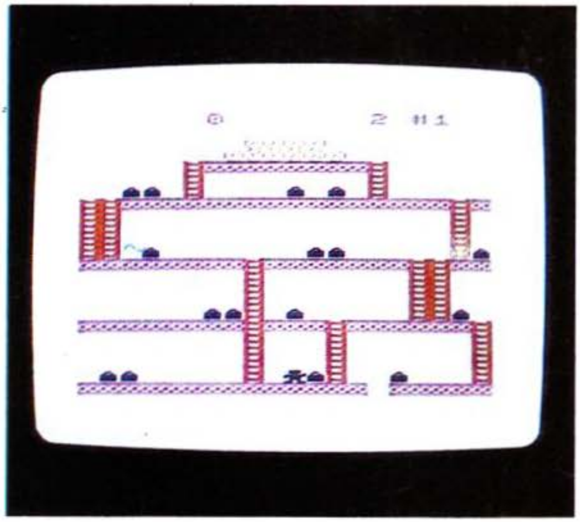
```
44 FORI=GTOG+24:POKEI,0:NEXT      :rem  30
45 POKEG+24,15:POKEG+5,17:POKEG+6,240:POK
   EG,100                         :rem 230
50 GOSUB710:E4=0:E5=2             :rem 192
60 GOSUB500:H=0:Y=0               :rem 106
70 S=1905+INT(RND(1)*38):IFPEEK(S+40)=62O
   RPEEK(S)=59THEN70              :rem 153
80 T=PEEK(S):POKES,58:POKES+G,0   :rem 161
90 V=1123+B(Y):W=62:DO=DO(INT(RND(1)*2))
                                  :rem 252
100 JV=255-PEEK(56321):GETQ$      :rem  61
101 IFJV=1THENJV=4:GOTO109        :rem 124
102 IFJV=4THENJV=3:GOTO109        :rem 127
103 IFJV=8THENJV=5:GOTO109        :rem 134
104 IFJV=2THEN109                 :rem 252
105 IFJVAND16=16THENJV=1:GOTO109:rem 237
108 JV=0                          :rem 166
109 ONJVGOTO210,120,140,170,190   :rem 173
110 FORN=1TO23:NEXT:GOTO260       :rem 193
120 IFPEEK(S+40)=ZTHENPOKES,T:POKES+G,D(T
    -J):S=S+40:GOTO250            :rem 178
130 GOTO260                       :rem 101
140 DI=-1:IFPEEK(S+39)<62THENPOKES,T:POKE
    S+G,D(T-J):S=S-1:GOTO250      :rem 250
150 IFT<>ZTHENPOKES,T:POKES+G,D(T-J):S=S+
    DI:T=PEEK(S):GOTO360          :rem 173
160 GOTO260                       :rem 104
170 IFT=ZTHENPOKES,T:POKES+G,D(T-J):S=S-4
    0:GOTO250                     :rem 181
180 GOTO260                       :rem 106
190 DI=1:IFPEEK(S+41)<62THENPOKES,T:POKES
    +G,D(T-J):S=S+1:GOTO250       :rem 201
200 GOTO150                       :rem  97
210 POKEG+1,17:POKEG+4,17         :rem 156
215 POKES,T:POKES+G,D(T-J):S=S-40+DI:T=PE
    EK(S):POKES,58:IFT=60THEN360:rem  43
220 IFPEEK(S+40)=60THENSS=SS+1000:PRINT"
    {HOME}{RVS}"TAB(8-LEN(STR$(SS)))SS
                                  :rem 105
230 FORN=1TO5:NEXT:POKES,T:POKES+G,D(T-J)
    :S=S+40+DI:T=PEEK(S):POKES,58
                                  :rem 204
235 IFS>1943ORPEEK(S+40)>61THEN360
                                  :rem 241
240 POKEG+4,16:POKES+G,0:GOTO260:rem 147
250 POKEG+1,9:POKEG+4,17:POKEG+4,16:T=PEE
    K(S):POKES,58:POKES+G,0       :rem 134
260 IFT<>61THEN270                :rem  36
265 SS=SS+150:PRINT"{HOME}{RVS}"TAB(8-LEN
    (STR$(SS)))SS:H=H+1:T=62:IFH=16THEN44
    0                             :rem 160
270 IFT=60THEN360                 :rem 231
280 GOSUB690                      :rem 185
290 POKEV,W:POKEV+G,D(W-J):V=V+DO:W=PEEK(
    V):POKEV,60:POKEV+G,7         :rem 195
300 IFDO=40ANDPEEK(V+40)=56THENDO=DO(INT(
    RND(1)*2)):GOTO320            :rem  21
310 IFW=63THENDO=40               :rem 131
320 IFW=58THEN360                 :rem 237
330 IFV<1944THEN100               :rem  73
340 Y=Y+1:IFY=12THEN60            :rem  57
350 POKEV,62:GOTO90               :rem 141
360 POKEG+4,16:SO=250:IFT=60THENT=W
                                  :rem 136
361 FORQQ=1TO30:NEXT              :rem  16
365 T1=T                          :rem 180
370 POKEG+1,SO:POKEG+4,33:IFPEEK(S+40)<>5
    6ANDS<1944THEN375             :rem 227
372 GOTO380                       :rem 112
375 POKES,T:POKES+G,D(T-J):S=S+40:T=PEEK(
    S):POKES,58:POKES+G,0         :rem 122
```



```
380 FORN=1TO17:NEXT:SO=SO-5:IFSO>150THEN3
    70                            :rem 231
390 POKEG+4,16:CH=CH-1:IFCH=-1THENPOKE532
    72,21:GOTO1000                :rem   2
400 PRINT"{HOME}{RVS}"TAB(25)CH:POKEV,W:P
    OKEV+G,D(W-J):Y=Y+1           :rem  43
405 IFW=58THENPOKEV,T1:POKEV+G,D(T1-J)
                                  :rem 124
410 IFY>10THEN60                  :rem 177
420 IFS>1943THENPOKES,T:GOTO70    :rem 146
430 POKES,58:POKES+G,0:GOTO90     :rem  24
440 IFY=11THEN470                 :rem 233
450 FORN=Y+1TO11:POKE1123+B(N),62:SS=SS+1
    00:PRINT"{HOME}{RVS}"TAB(8-LEN(STR$(S
    S)))SS                        :rem 187
455 GOSUB690:NEXT                 :rem  54
460 POKEG+4,33:FORM=9TO176STEP2:POKEG+1,M
    :NEXT:POKEG+4,32              :rem  50
470 E2=E2+.05:SC=SC+1:E1=E1+1:IFE1>8THENE
    1=8                           :rem  17
480 GOTO60                        :rem  59
490 GOTO490                       :rem 115
500 PRINT"{CLR}{PUR}";:FORN=1TO22:PRINTA$
    :NEXT:PRINTA$"{HOME}"         :rem 220
505 B$=">88888888888888888888888888888888
    888888"                       :rem  15
510 PRINT"{ 3 GIU'}"TAB(15)"?>>>>>>>>?
    {RED}":PRINTTAB(15)"9{PUR}88888888
    {RED}9"                       :rem 121
515 PRINTTAB(15)"9>>>>>>>>9":PRINTTAB(15)
    "9>>>>>>>>9{PUR}"             :rem 243
520 FORN=1TO3:PRINTB$"{ 4 GIU'}":NEXT:PRI
    NTB$"{HOME}";:POKE1983,62     :rem 207
530 PRINT"{RVS}"TAB(8-LEN(STR$(SS)))SS;TA
    B(25)CH;TAB(31)SC:POKE1054,163
                                  :rem 157
535 FORN=0TO11:POKE1123+B(N),60 :rem  18
540 POKE1123+B(N)+G,7:NEXT:FORN=1344TO194
    4STEP200:IFN=1944THEN600      :rem 143
550 FORO=1TO3                     :rem  20
560 R=N+1+INT(RND(1)*38):IFPEEK(R)<>56THE
    N560                          :rem 118
570 FORM=RTOR+160STEP40:POKEM,57:POKEM+G,
    2:NEXT                        :rem   3
575 IFO>1ANDRND(1)<E2THENPOKER+(INT(RND(1
    )*2)+2)*40,63                 :rem 132
```

```
580 IFRND(1)<.5ANDPEEK(R-40)=62THENPOKER-
    40,63                          :rem 216
590 NEXT                           :rem 221
600 FORO=1TOE1                     :rem  83
610 R=N+3+INT(RND(1)*36)           :rem  50
615 IFPEEK(R)<>56ORPEEK(R-40)<>62ORPEEK(R
    +1)=62ORPEEK(R-1)=62THEN630 :rem  82

620 POKER,62:POKER-40,63           :rem 142
630 NEXT                           :rem 216
640 FORO=1TO4                      :rem  21
650 R=N-39+INT(RND(1)*38):IFPEEK(R)<>62OR
    PEEK(R+40)=62THEN650           :rem  77
660 POKER,61:POKER+G,0:NEXT:NEXT:rem  43
670 POKE1122,63:POKE1125,63:POKE1161,63:P
    OKE1166,63                     :rem 129
680 FORN=1304TO1904STEP200:POKEN,63:NEXT:
    FORN=1343TO1943STEP200:POKEN,63:NEXT
                                   :rem  90
685 RETURN                         :rem 131
690 IFSS>=Q*E3THENCH=CH+1:E3=E3+1:PRINT"
    {HOME}{RVS}"TAB(25)CH          :rem  45
700 RETURN                         :rem 119
710 DATA,1,39,40,41,42,78,79,80,81,82,83
                                   :rem 231
720 DATA255,255,153,102,102,153,255,255,1
    95,255,255,195,195,255,255,195,60,60
                                   :rem  31
725 DATA25,255,188                 :rem 180
730 DATA60,36,231,3,4,24,24,60,126,126,60
    ,60,66,165,153,153,165,66,60,,24,36
                                   :rem 195
735 DATA126,126,126                :rem 220
740 DATA126,,,,,,,,,,,,,,,,,       :rem  58
750 DATA169,,133,1,169,255,141,34,145,169
    ,32,44,31,145,208,5,169,1,133,1,96
                                   :rem 163
755 DATA169,8,44                   :rem  83
760 DATA31,145,208,5,169,2,133,1,96,169,1
    6,44,31,145,208,5,169,3,133,1,96,169
                                   :rem  15
765 DATA4,44,31                    :rem  20
770 DATA145,208,3,133,1,96,169,127,141,34
    ,145,169,128,44,32,145,208,4,169,5
                                   :rem 176
775 DATA133,1,96                   :rem  76
776 FORN=0TO11:READB(N):NEXT       :rem 251
777 RETURN                         :rem 133
780 FORI=0TO11:READX:NEXT:PRINT"
    { 9 GIU'}{ 4 DES}COSTRUZIONE DEL CAMP
    O DI GIOCO"                    :rem  40
782 POKE56334,PEEK(56334)AND254:POKE1,PEE
    K(1)AND251                     :rem 194
783 FORI=12288TO12288+256*8:POKEI,PEEK(I+
    40960):NEXTI                   :rem 115
784 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334
    )OR1                           :rem 146
785 FORI=12288+56*8TO12288+63*8+7:READM:P
    OKEI,M:NEXTI                   :rem 158
786 POKE53272,(PEEK(53272)AND240)+12
                                   :rem 196
800 RETURN                         :rem 120
1000 PRINT"{CLR}IL TUO PUNTEGGIO E' ";SS
                                   :rem 177
1010 PRINT"{ 6 GIU'}VUOI GIOCARE ANCORA (
     S/N) ?";                      :rem 133
1020 GETA$:IFA$=""THEN1020         :rem 169
1030 IFA$="S"THENPOKE53272,(PEEK(53272)AN
     D240)+12:GOTO2                :rem 127
1040 IFA$<>"N"THEN1020             :rem 183
1050 SYS2048                       :rem 147
```



Lo scalatore versione per VIC 20

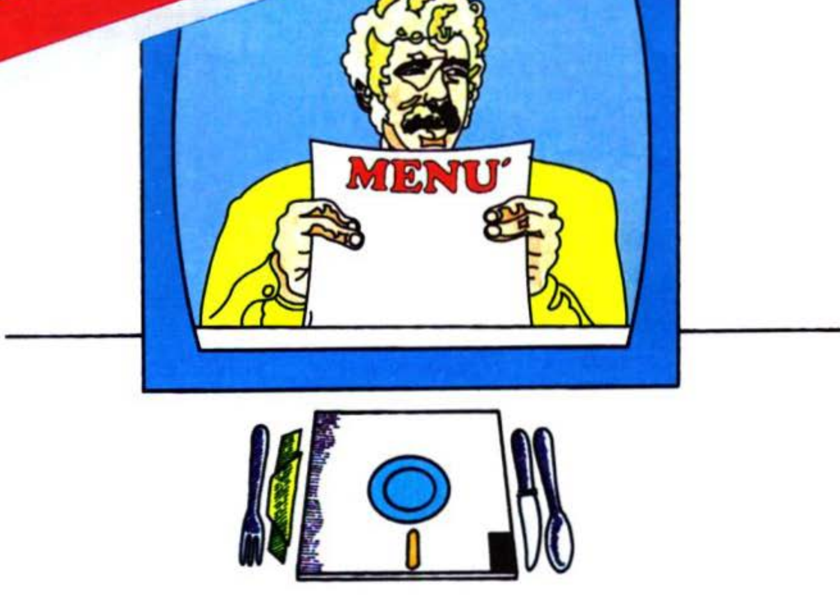
```
0 PRINT"{CLR}":POKE51,192:POKE52,29:POKE5
  5,192:POKE56,29:POKE36869,255:POKE36878
  ,15:POKE36879,25                 :rem 241
1 D=37154:P1=37151:P2=37152:DO(0)=-1:DO(1
  )=1:DI=DO(INT(RND(1)*2))         :rem  54
2 A$=">>>>>>>>>>>>>>>>>>>>>{SIN}{INST}>":
  DIMB(11):G=30720:E2=0            :rem 126
3 SC=1:CH=2:E1=0:D(0)=4:D(1)=2:D(4)=7:Z=5
  7:E3=1:Q=10000:J=56              :rem  32
10 GOSUB100:POKE36869,255:E4=0:E5=2
                                   :rem 246
15 GOSUB70:H=0:Y=0                 :rem  60
16 S=8143+INT(RND(1)*20):IFPEEK(S+22)=620
   RPEEK(S)=59THEN16               :rem 145
17 T=PEEK(S):POKES,58:POKES+G,0 :rem 161
19 V=7712+B(Y):W=62:DO=DO(INT(RND(1)*2))
                                   :rem   7
20 SYS828:ONPEEK(1)GOTO35,26,28,31,33
                                   :rem 110
21 FORN=1TO23:NEXT:GOTO41          :rem  95
26 IFPEEK(S+22)=ZTHENPOKES,T:POKES+G,D(T-
   J):S=S+22:GOTO40                :rem  84
27 GOTO41                          :rem   7
28 DI=-1:IFPEEK(S+21)<62THENPOKES,T:POKES
   +G,D(T-J):S=S-1:GOTO40          :rem 147
29 IFT<>ZTHENPOKES,T:POKES+G,D(T-J):S=S+D
   I:T=PEEK(S):GOTO55              :rem  83
30 GOTO41                          :rem   1
31 IFT=ZTHENPOKES,T:POKES+G,D(T-J):S=S-22
   :GOTO40                         :rem  78
32 GOTO41                          :rem   3
33 DI=1:IFPEEK(S+23)<62THENPOKES,T:POKES+
   G,D(T-J):S=S+1:GOTO40           :rem  98
34 GOTO29                          :rem  11
35 POKE36876,240:POKES,T:POKES+G,D(T-J):S
   =S-22+DI:T=PEEK(S):POKES,58:IFT=60THEN
   55                              :rem   5
36 IFPEEK(S+22)=60THENSS=SS+1000:PRINT"
   {HOME}{RVS}"TAB(8-LEN(STR$(SS)))SS
                                   :rem  62
37 FORN=1TO5:NEXT:POKES,T:POKES+G,D(T-J):
   S=S+22+DI:T=PEEK(S):POKES,58:IFPEEK(S+
   22)>61THEN55                    :rem   0
38 POKES+G,0:POKE36876,0:GOTO41 :rem 102
```

```
40 POKE36876,200:POKE36876,0:T=PEEK(S):PO
   KES,58:POKES+G,0                 :rem 165
41 IFT=61THENSS=SS+150:PRINT"{HOME}{RVS}"
   TAB(8-LEN(STR$(SS)))SS:H=H+1:T=62:IFH=
   16THEN64                         :rem 240
42 IFT=60THEN55                     :rem 133
43 GOSUB98                          :rem  88
45 POKEV,W:POKEV+G,D(W-J):V=V+DO:W=PEEK(V
   ):POKEV,60:POKEV+G,7             :rem 145
46 IFDO=22ANDPEEK(V+22)=56THENDO=DO(INT(R
   ND(1)*2)):GOTO48                 :rem 195
47 IFW=63THENDO=22                  :rem  90
48 IFW=58THEN55                     :rem 149
49 IFV<8164THEN20                   :rem 242
50 Y=Y+1:IFY=12THEN55               :rem  11
51 POKEV,62:GOTO19                  :rem  92
55 POKE36876,0:SO=250:IFT=60THENT=W
                                    :rem 138
56 POKE36874,SO:IFPEEK(S+22)<>56ANDS<8164
   THENPOKES,T:POKES+G,D(T-J):S=S+22:T=PE
   EK(S):POKES,58:POKES+G,0         :rem 185
57 FORN=1TO17:NEXT:SO=SO-5:IFSO>150THEN56
                                    :rem 137
58 POKE36874,0:CH=CH-1:IFCH=-1THENPOKED,2
   55:POKE36869,240:PRINT"{CLR}{BLK}"SS:E
   ND                               :rem 104
59 PRINT"{HOME}{RVS}"TAB(14)CH::POKEV,W:P
   OKEV+G,D(W-J):Y=Y+1:IFW=58THENPOKEV,T:
   POKEV+G,D(T-J)                   :rem 248
60 IFY>10THEN15                     :rem 130
61 IFS>8163THENPOKES,T:GOTO16       :rem 100
62 POKES,58:POKES+G,0:GOTO19        :rem 234
64 IFY=11THEN67                     :rem 141
65 FORN=Y+1TO11:POKE7712+B(N),62:SS=SS+10
   0:PRINT"{HOME}{RVS}"TAB(8-LEN(STR$(SS)
   ))SS:GOSUB98                     :rem 194
66 POKE36877,250:FORM=240TO250:POKE36876,
   M:NEXT:POKE36876,0:POKE36877,0:NEXT
                                    :rem 113
67 E2=E2+.05:SC=SC+1:E1=E1+1:IFE1>8THENE1
   =8                               :rem 227
68 GOTO15                           :rem  13
69 GOTO69                           :rem  23
70 PRINT"{CLR}{PUR}";:FORN=1TO21:PRINTA$:
   NEXT:PRINTA$"{HOME}":B$=">888888888888
   88888888"                        :rem 108
71 PRINT"{ 2 GIU'}"TAB(6)"?>>>>>>>>?{RED}
   ":PRINTTAB(6)"9{PUR}88888888{RED}9":PR
   INTTAB(6)"9>>>>>>>>9":PRINTTAB(6)"9>>>
   >>>>>9{PUR}"                     :rem  12
72 FORN=1TO3:PRINTB$"{ 4 GIU'}":NEXT:PRIN
   TB$"{HOME}";:POKE8185,62         :rem 162
73 PRINT"{RVS}"TAB(8-LEN(STR$(SS)))SS;TAB
   (14)CH;TAB(17)SC:POKE7697,163:FORN=0TO
   11:POKE7712+B(N),60              :rem  61
74 POKE7712+B(N)+G,7:NEXT:FORN=7834TO8164
   STEP110:IFN=8164THEN80           :rem  73
75 FORO=1TO3                        :rem 230
76 R=N+1+INT(RND(1)*20):IFPEEK(R)<>56THEN
   76                               :rem  17
77 FORM=RTOR+88STEP22:POKEM,57:POKEM+G,2:
   NEXT:IFO>1ANDRND(1)<E2THENPOKER+(INT(R
   ND(1)*2)+2)*22,63                :rem 203
78 IFRND(1)<.5ANDPEEK(R-22)=62THENPOKER-2
   2,63                             :rem 170
79 NEXT                             :rem 175
80 FORO=1TOE1                       :rem  37
81 R=N+3+INT(RND(1)*16):IFPEEK(R)<>56ORPE
   EK(R-22)<>62ORPEEK(R+1)=62ORPEEK(R-1)=
   62THEN85                         :rem 198
84 POKER,62:POKER-22,63             :rem  98
85 NEXT                             :rem 172
86 FORO=1TO4                        :rem 233
87 R=N-21+INT(RND(1)*20):IFPEEK(R)<>62ORP
   EEK(R+22)=62THEN87               :rem 227
88 POKER,61:POKER+G,0:NEXT:NEXT     :rem 255
89 POKE7710,63:POKE7715,63:POKE7731,63:PO
   KE7738,63                        :rem 125
90 FORN=7812TO8142STEP110:POKEN,63:NEXT:F
   ORN=7833TO8163STEP110:POKEN,63:NEXT:RE
   TURN                             :rem  85
98 IFSS>=Q*E3THENCH=CH+1:E3=E3+1:PRINT"
   {HOME}{RVS}"TAB(14)CH            :rem 253
99 RETURN                           :rem  82
100 DATA,1,21,22,23,24,42,43,44,45,46,47
                                    :rem 197
101 DATA255,255,153,102,102,153,255,255,1
    95,255,255,195,195,255,255,195,60,60,
    25,255,188                      :rem  64
102 DATA60,36,231,3,4,24,24,60,126,126,60
    ,60,66,165,153,153,165,66,60,,24,36,1
    26,126,126                      :rem  11
103 DATA126,,,,,,,,,,,,,,,,         :rem  51
106 DATA169,,133,1,169,255,141,34,145,169
    ,32,44,31,145,208,5,169,1,133,1,96,16
    9,8,44                          :rem  98
107 DATA31,145,208,5,169,2,133,1,96,169,1
    6,44,31,145,208,5,169,3,133,1,96,169,
    4,44,31                         :rem 142
108 DATA145,208,3,133,1,96,169,127,141,34
    ,145,169,128,44,32,145,208,4,169,5,13
    3,1,96                          :rem 102
109 FORN=0TO11:READB(N):NEXT:FORN=7616TO7
    679:READM:POKEN,M:NEXT          :rem  63
110 FORN=828TO899:READM:POKEN,M:NEXT:RETU
    RN                              :rem  89
```

NON PERDETE IL PROSSIMO NUMERO! IN EDICOLA TRA UN MESE

Disk menu

di Disk Menu
trad. e adatt. F. Stella



Su molti computer sono disponibili utility per caricare e lanciare programmi su disco con un solo comando, eccone finalmente uno anche per VIC 20 e Commodore 64.

I disk drive 1540 e 1541 sono stati una benedizione per tutti gli utenti dei piccoli Commodore quanto a risparmio di tempo ma, quando vi è un gran numero di programmi su un solo disco, il caricare e listare la directory per cercare un certo programma è comunque una faccenda lenta e macchinosa. Originariamente quésto programma doveva servire ad ottenere una migliore visualizzazione della directory ma poi, aggiungervi le appendici AUTOLOAD e AUTORUN, cioè automatizzare queste operazioni, è stata una logica conseguenza. Il programma è facile da usare e si spiega da sè.

Come funziona

La routine di caricamento della directory (che parte alla riga 160) apre il file "$" e poi lo legge, carattere per carattere, sino a trovare le virgolette; la prima sequenza di caratteri incontrata è il nome del disco, ognuna delle successive è un nome di file; tutte queste informazioni vengono caricate nel vettore P$. Per caricare ed eventualmente lanciare i programmi viene usata la tecnica di simulazione di input da tastiera; viene visualizzato un comando LOAD e poi il cursore è riportato in su così da posizionarvisi sopra. Il buffer della tastiera viene caricato con un CHR$(13) (RETURN) oppure, nell'opzione autorun con una sequenza RETURN-RUN-RETURN; quando l'interprete incontra la END svuota il buffer ed esegue i comandi diretti predisposti nella coda in sequenza opportuna.

La linea 510 accorcia i nomi dei file per evitare che superino in lunghezza una riga di schermo (è assolutamente superflua per il C 64).

DISK MENU non discrimina tra programmi ed altri tipi di file; se ciò dovesse essere ritenuto un difetto, basterà modificare la routine di caricamento della directory facendole analizzare il byte 0 di ogni intestazione di file e considerando poi solo quelli che contengono un 2 (tipo PRG).

Dato che le locazioni utilizzate per la gestione della tastiera sono le stesse per il VIC 20 ed il C 64, il programma funziona altrettanto bene su entrambi. Deve essere cancellata per l'uso con VIC 20 alla linea 110 la POKE 53281, 1 che serve nel C 64 a cambiare il colore del fondo e visualizzare meglio i caratteri.

Il menù può contenere fino a 100 programmi (già molti per un disco!); se questo numero dovesse essere insufficiente, bisognerà variare il valore 100 nella DIM P$ ed il limite posto nell'istruzione alla riga 330. L'autorun può essere validamente sfruttato per concatenare due o più programmi fra loro. SE **DISK MENU** è salvato come primo programma di un disco può poi essere caricato con il comando LOAD"*",8 che uniforma e semplifica ulteriormente l'operazione.

Disk Menu versione per VIC 20 e C 64

```
110 POKE53281,1:CLR:DIMP$(100)   :rem  166
130 PRINT"{CLR}{ 3 GIU'}{GRN} STO LEGGEND
    O LA DIRECTORY{BLU}"         :rem  229
170 OPEN1,8,0,"$"                :rem   79
190 GET#1,B$:IFST<>0THEN270      :rem   66
200 IFB$<>CHR$(34)THEN190        :rem  131
210 P$=""                        :rem  136
220 GET#1,B$:IFB$<>CHR$(34)THENP$=P$+B$:G
    OTO220                       :rem  168
225 IFP$="{ 2 SPAZI}"THENP$="":GOTO220
                                 :rem   72
230 GET#1,B$:IFB$=CHR$(32)THEN230
                                 :rem   66
240 P$(X)=P$:PRINTX"{SU}"        :rem   88
250 GET#1,B$:IFB$<>""THEN250     :rem   16
260 IFST=0THENX=X+1:GOTO190      :rem  192
270 CLOSE1:N=10                  :rem  102
```

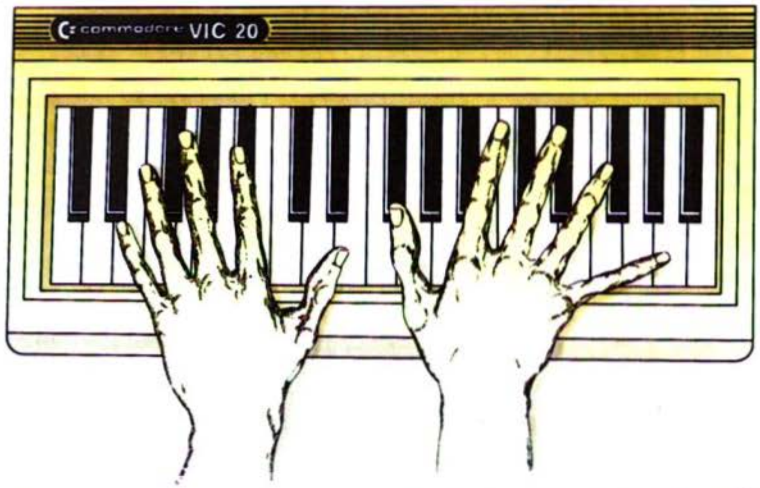
```
300 PRINT"{CLR}{RED} DISK MENU{GIU'}"
                                :rem  132
310 PRINT" DI{ 2 SPAZI}";P$(0)"{BLU}":PRI
    NT                          :rem   76
320 IFN<10THENN=10              :rem   42
330 IFN>100THENN=100            :rem  141
340 FORJ=(N-9)TON:PRINTJ"- "P$(J):NEXT
                                :rem    5
350 PRINT:PRINT"{GIU'}{GRN}{ 5 SPAZI}PREM
    ERE{GIU'}":PRINT"{ 5 SPAZI}{RED}N
    {BLU} PER PROSEGUIRE"       :rem    0
360 PRINT"{ 5 SPAZI}{RED}L{BLU} PER TORNA
    RE INDIETRO"                :rem   81
370 PRINT"{ 5 SPAZI}{RED}E{BLU} PER CHIUD
    ERE"                        :rem   27
380 PRINT"{ 5 SPAZI}{RED}S{BLU} PER ESEGU
    IRE IL LOAD"                :rem  239
390 PRINT"{ 5 SPAZI}{RED}R{BLU} PER LOAD+
    RUN"                        :rem   33
400 Z$="":GETZ$:IFZ$=""THEN400  :rem  182
410 IFZ$="N"THENN=N+10:GOTO300  :rem   11
420 IFZ$="L"THENN=N-10:GOTO300  :rem   12
430 IFZ$="E"THENEND             :rem  112
440 IFZ$="S"THENGOSUB470:GOTO570:rem  210
450 IFZ$="R"THENGOSUB470:GOTO540:rem  207
460 GOTO400                     :rem  103
470 REM SELEZIONA E STAMPA LOAD :rem   84
480 PRINT"{GIU'} PROGRAMMA N."; :rem  219
490 INPUTS:PRINT"{ 17 GIU'}"    :rem  172
500 IFS<1ORS>XTHEN480           :rem   57
510 IFLEN(P$(S))>12THENP$(S)=LEFT$(P$(S),
    12)+"*"                     :rem  114
520 PRINT"LOAD"+CHR$(34)+P$(S)+CHR$(34)+"
    ,8{ 3 SU}":RETURN           :rem   51
540 REM LOAD+RUN                :rem  189
550 POKE631,13:POKE632,82:POKE633,85:POKE
    634,78:POKE635,13:POKE198,5 :rem  207
560 END                         :rem  114
570 REM SOLO LOAD               :rem  221
580 POKE631,13:POKE198,1        :rem   94
```





Far musica con il computer

di C. Regena
trad. e adatt. U. Barzaghi



Usare il vostro calcolatore per far musica può essere molto divertente. Potete far "girare" un programma che faccia suonare al vostro calcolatore un motivo, quindi potete cantare seguendo il calcolatore. Se suonate uno strumento musicale come solista, potete programmare il calcolatore perché vi accompagni. Se state imparando a suonare un brano musicale particolarmente difficile, potete far suonare il brano al calcolatore in modo da poter capire come il brano dovrebbe essere suonato. Video-giochi possono essere arricchiti con intermezzi musicali. I programmi educativi possono usare la musica per segnalare le risposte esatte. Insegnanti e studenti di musica possono usare programmi musicali per migliorare il processo di apprendimento. Ogni tipo di esercitazione ripetitiva può essere svolta con l'aiuto del computer.

Per programmare musica sul vostro computer, dovete prendere in considerazione il volume (intensità), la nota (tono), e la durata (lunghezza di una nota). Il VIC 20 dispone di tre "canali" o "voci" per suonare che possono essere utilizzati contemporaneamente, più una voce per i rumori.

Calcolatori differenti hanno diversi comandi in BASIC per produrre musica. Altri micro-calcolatori possono usare comandi come SOUND e PLAY.

Nella programmazione del VIC viene utilizzato il comando POKE. L'istruzione POKE può creare confusione perché la sua forma è POKE **nl, n2**, dove **nl, n2** sono numeri e bisogna sapere che genere di valori numerici usare. **Nl** è una "locazione di memoria" o "indirizzo", per il VIC un numero compreso fra 0 e 65535. **N2** è un valore numerico che voi inserite nella locazione di memoria suddetta.

Per il VIC dev'essere un numero tra 0 e 255. Il calcolatore opererà in base ai numeri da voi utilizzati. Alcuni comandi di POKE fanno riferimento alla grafica; altri possono modificare caratteristiche del calcolatore come la velocità o disabilitare determinati tasti.

Volume

Per regolare il volume in modo da poter suonare, il valore numerico della locazione di memoria è 36878, ed il livello del volume può andare da 0 (spento) a 15 (massimo). Il volume è normalmente spento, quindi se vi capitasse di dimenticare un comando simile all'interno del vostro programma non sentireste alcuna musica. POKE 36878,15 regola il volume al massimo livello.

Potreste desiderare di utilizzare i valori numerici suddetti come variabili:

```
100 V=36878
110 L=15
120 POKE V,L
•
•
•
300 L=10
310 POKE V,L
•
•
•
500 POKE V,0
```

Nota o tono

Anche se l'altoparlante è attivato, non sentireste niente finché non comanderete al calcolatore di suonare una nota. Ci sono tre locazioni che possono essere utilizzate per i canali sonori: 36874, 36875 e 36876.

Ogni voce ha un differente campo di toni disponibile. 36874 ha le note più alte. I numeri che potete inserire con una istruzione di POKE in queste locazioni di memoria per generare suoni possono occupare un campo di valori da 128 a 255, dove 255 è la nota più alta in ciascun campo. Ci sono delle tabelle nel manuale che viene venduto con il vostro calcolatore e nel "VIC 20 Programmer's Reference Guide" che traduce ogni nota nel valore numerico necessario per essere utilizzato dal linguaggio del calcolatore.

Proviamo qualche nota. Il breve programma seguente serve ad illustrare i comandi necessari per suonare una singola nota:

```
100 POKE 36878,10
110 POKE 36874,183
120 END
```

Mandate in esecuzione il programma ed udrete la nota LA. Potrete notare che, anche se il programma termina, continuerete a sentire la nota. Dovete eseguire l'istruzione POKE 36874,0 per porre fine al suono. Potete anche spegnere l'altoparlante mediante l'istruzione POKE 36878,0, o premendo il tasto RUN/STOP contemporaneamente a RESTORE.

Durata

Il calcolatore continua a suonare una nota finché non cambiate la nota o non dite al calcolatore di fermarsi. Il vostro programma può eseguire calcoli o disegnare figure mentre il calcolatore sta suonando (in questo modo gli effetti sonori vengono sincronizzati con la grafica nei videogiochi). Ci sono parecchi modi per far suonare una nota al calcolatore per un certo intervallo di tempo e quindi passare ad una nota differente o fermarsi.
Un metodo di temporizzazione consiste nell'usare un ciclo FOR-NEXT.

Un ciclo ha il seguente aspetto:

```
20 FOR D=1 TO 10
25 PRINT D
30 NEXT D
```

L'istruzione FOR comanda al calcolatore di effettuare il ciclo la prima volta con D=1, la volta successiva con D=2, la volta successiva con D=3, e così via sinché D è maggiore del limite 10 (l'ultimo ciclo verrà effettuato con D=10). I cicli FOR-NEXT sono un modo di far sì che il calcolatore ripeta qualcosa più volte con solo poche righe di istruzione.
Ora, se voi cancellate la linea 25 del programma precedente avrete un ciclo FOR-NEXT senza comandi all'interno, quindi il calcolatore si limita a contare da uno a 10. Se cambiate il 10 in 100, noterete che il ciclo impiegherà un tempo maggiore. Potete ottenere differenti intervalli di tempo modificando il valore numerico del limite superiore del ciclo. Ecco come i cicli possono controllare la durata dei suoni:

```
10 POKE 36878,12
20 POKE 36875,207
30 FOR D=1 TO 300:NEXT D
40 POKE 36875,201
50 FOR D=1 TO 300:NEXT D
60 POKE 36875,195
70 FOR D=1 TO 600:NEXT D
80 POKE 36875,0
90 POKE 36878,0
100 END
```

Ricordatevi che nella programmazione ci sono più modi di ottenere la stessa cosa. Supponiamo di voler continuare il motivo nel programma di cui sopra e che la maggior parte delle note usi un ciclo di temporizzazione di 300. Possiamo utilizzare una istruzione di GOSUB per risparmiare qualche riga e un po' di spazio in memoria. GOSUB indica al calcolatore di andare (GO) ad un sottoprogramma (SUBroutine), (un breve programma all'interno del programma principale per eseguire alcune righe di istruzione), quindi rientrare (RETURN) all'istruzione seguente quella di GOSUB.
Ogni volta in cui effettuate la stessa azione un certo numero di volte può essere conveniente farne un sottoprogramma ed utilizzare un'istruzione GOSUB.
Ecco un esempio:




```
10 POKE 36878,12
20 POKE 36875,207
30 GOSUB 200
40 POKE 36875,201
50 GOSUB 200
60 POKE 36875,207
70 GOSUB 200
80 POKE 36875,215
90 GOSUB 200
•
•
•
180 POKE 36875,0
190 POKE 36878,0
199 END
200 FOR D=1 TO 300
210 NEXT D
220 RETURN
```

Per risparmiare linee di programma (quindi spazio in memoria), potete porre S=36875. Quindi ogni volta che avete bisogno di utilizzare il valore numerico 36875, sostituitegli la variabile S. Nel BASIC Commodore voi potete anche tralasciare le variabili nei comandi NEXT, e combinare più righe tramite la loro separazione con i due punti (:). Potete anche abbreviare il comando di POKE battendo P, seguito da SHIFT O. Potete anche tralasciare gli spazi per risparmiare memoria.
Il seguente programma illustra un modo di utilizzare i sottoprogrammi ed i cicli FOR-NEXT per note di diversa lunghezza. GOSUB 56 rappresenta la lunghezza necessaria per una nota di una ottava. GOSUB 54 è lunga il doppio e rappresenta una nota di una quarta; GOSUB 52 è una quarta puntata (uguale ad una nota lunga tre ottavi); e GOSUB 50 è usata per una mezza nota.

```
1  REM MELODIA
2  POKE 36878,15
3  S=36876
4  T=200
5  POKES,195:GOSUB 52
7  POKES,187:GOSUB 56
9  POKES,183:GOSUB 54
11 POKES,175:GOSUB 54
13 POKES,163:GOSUB 54
15 POKES,175:GOSUB 54
17 POKES,183:GOSUB 54
19 POKES,163:GOSUB 54
21 POKES,175:GOSUB 56
23 POKES,183:GOSUB 56
25 POKES,187:GOSUB 56
```

```
27 POKES,175:GOSUB 56
29 POKES,183:GOSUB 52
31 POKES,175:GOSUB 56
33 POKES,163:GOSUB 54
35 POKES,159:GOSUB 54
37 POKES,163:GOSUB 50
39 POKES,0:POKE 36878,0
41 END
50 FOR D=1 TO T:NEXT
52 FOR D=1 TO T:NEXT
54 FOR D=1 TO T:NEXT
56 FOR D=1 TO T:NEXT:RETURN
60 END
```

Notate che la riga 4 inizializza T=200. Se volete cambiare il tempo di questo motivo, tutto ciò che è necessario fare consiste nel cambiare questa riga. Per esempio, per suonare la melodia più lentamente, cambiare la riga 4 in T=350 e mandare il programma in esecuzione. Ora cambiate la riga 4 in T=100 e battete RUN. La durata delle varie note rimane nelle opportune proporzioni, ma la velocità è diversa.
Il programma successivo suona la stessa melodia, ma usando delle istruzioni DATA per leggere (READ) i valori delle note ed i fattori di durata. La riga 4 inizializza ancora una variabile T in modo da poter cambiare il tempo.

```
1  REM MELODIA 2
2  POKE 36878,15
3  S=36878
4  T=200
5  READ N,A
7  IF N= —1 THEN 39
9  POKE S,N
11 FOR D=1 TO T*A:NEXT
13 GOTO 5
15 DATA 195,3,187,1,183,2,175,2,
16,3,2, 175,7,183,2,163,2,175,1
17 DATA 183,1,187,1,175,1,183,3,
17,5,1,163,2,159,2,163,4,—1,—1
39 POKE S,0:POKE 36878,0
41 END
```

Il modo migliore di conoscere le possibilità offerte dal vostro calcolatore circa la generazione dei suoni, consiste nello scrivere i vostri programmi. Fate esperimenti con il volume per porre l'accento su certe note. Provate differenti durate per sviluppare ritmi complessi. Provate ad utilizzare diverse combinazioni numeriche con le tre differenti voci per sentire il campo di variabilità dei toni, provate ad usare più di una voce contemporaneamente per creare una armonia.

LOAD

Far musica con il computer

Impariamo le note: la tastiera

Questo programma, "la tastiera", è studiato per gli allievi di piano oppure organo alle prime armi che imparano i nomi delle note sulla tastiera. Un insegnante di musica può utilizzare questo programma prima o dopo una normale lezione.
Il programma presenta i nomi delle note corrispondenti ai tasti di un piano o di un organo. Dopo alcune pagine video descrittive, viene proposto un quiz consistente in dieci tasti. Viene scelta casualmente una serie di tasti, tra una serie che inizia con due tasti neri ed una serie che inizia con tre tasti neri. Un asterisco rosso appare casualmente su uno dei tasti. Lo studente deve indicare il nome della nota corrispondente al tasto indicato. Se la risposta è errata, viene prodotto un breve rumore e lo studente deve riprovare.
Questo programma utilizza la grafica a scopi educativi. Le possibilità musicali del calcolatore permettono di suonare la nota corrispondente in modo che lo studente possa sentire il suono e vedere fisicamente la posizione del tasto corrispondente.

Linee	Commenti
2	Mette il volume al massimo livello; compie una diramazione ad una subroutine successiva.
3-16	Sottoprogramma per le note. **AA$** è la stringa che rappresenta il nome della nota. **S** è il numero corrispondente al tono. **CC** è l'indice di colonna del tasto corrispondente sulla tastiera. La variabile **K** può avere valore 1 o 2, a seconda del tipo di tastiera.
20-23	Sottoprogrammi grafici, utilizzati per disegnare le tastiere. I comandi di RVS ON e RVS OFF vengono utilizzati per ottenere i tasti bianchi e neri.
50-51	Stampa il titolo. La variabile **SS** contiene il valore numerico associato all'altoparlante (36876).
52-54	Utilizzando i valori numerici contenuti nelle frasi DATA, suona una scala musicale, quindi si salta alla sezione educativa.
55	Propone un quiz composto da dieci tasti.
56	Sceglie casualmente, e quindi disegna, una delle due tastiere.
58-62	Sceglie casualmente una nota **N**. In base alla tastiera ed alla nota scelta, salta ad una subroutine che consente di stabilire il nome della nota, il tono e la coordinata della posizione corrispondente sulla tastiera.
64-66	Fa lampeggiare un asterisco rosso sul tasto scelto.
68	Acquisisce la nota digitata dallo studente sulla tastiera.
70-72	Se la nota non è corretta, viene prodotto un rumore e si ricicla per un'altra risposta.
79-80	Se la nota è corretta, ne stampa il nome sul tasto, ne suona il tono, quindi salta ad una nuova domanda.
90-94	Offre la possibilità di sostenere un'altra serie di quiz.
104-106	Sottoprogrammi grafici per disegnare i tasti.
108-111	Sottoprogramma per stampare il messaggio "PREMI RETURN" su di una pagina video di istruzioni ed attendere la risposta dello studente.
150-152	Stampa della prima pagina video di istruzioni.
154-160	Stampa della seconda pagina video di istruzioni.
164-168	Stampa della terza pagina video di istruzioni; salto alla sezione che propone i quiz.
170	Fine.
200-220	Sottoprogramma di servizio per compitare i nomi delle note.

Tastiera
versione per VIC 20

```
2 POKE36878,15:GOTO50                    :rem 219
3 AA$="DO":S=131:CC=0:RETURN             :rem 147
4 AA$="RE":S=145:CC=3:RETURN             :rem 160
5 AA$="MI":S=158:CC=6:RETURN             :rem 167
6 AA$="FA":S=161:IFK=1THENCC=8:RETURN
                                         :rem  12
7 CC=0:RETURN                            :rem  68
8 AA$="SOL":S=173:IFK=1THENCC=11:RETURN
                                         :rem 162
9 CC=3:RETURN                            :rem  73
10 AA$="LA":S=181:IFK=1THENCC=14:RETURN
                                         :rem 108
11 CC=6:RETURN                           :rem 117
12 AA$="SI":S=189:IFK=1THENCC=17:RETURN
                                         :rem 136
13 CC=9:RETURN                           :rem 122
14 AA$="DO":S=192:CC=11:RETURN           :rem 254
15 AA$="RE":S=200:CC=14:RETURN           :rem 252
16 AA$="MI":S=206:CC=17:RETURN           :rem   5
20 FORKI=1TO10:PRINT" {BLK}{RVS}
   { 2 SPAZI}{OFF} {RVS}{ 2 SPAZI}{OFF}
   B {RVS}{ 2 SPAZI}{OFF} {RVS}
   T 2 SPAZI}{OFF} {RVS}{ 2 SPAZI}{OFF}
   B":NEXT                               :rem  27
21 FORKI=1TO5:PRINT"{ 2 SPAZI}[<G>]
   { 2 SPAZI}[<G>] B{ 2 SPAZI}[<G>]
   { 2 SPAZI}[<G>]{ 2 SPAZI}[<G>] B":NEXT
   :RETURN                               :rem 127
22 FORKI=1TO10:PRINT" {BLK}{RVS}
   { 2 SPAZI}{OFF} {RVS}{ 2 SPAZI}{OFF}
   {RVS}{ 2 SPAZI}{OFF} B {RVS}
   { 2 SPAZI}{OFF} {RVS}T 2 SPAZI}{OFF}
   B":NEXT                               :rem  29
23 FORKI=1TO5:PRINT"{ 2 SPAZI}[<G>]
   { 2 SPAZI}[<G>]{ 2 SPAZI}[<G>] B
   { 2 SPAZI}[<G>]{ 2 SPAZI}[<G>] B":NEXT
   :RETURN                               :rem 129
50 PRINT"{CLR}{BLU}{ 3 GIU'}{ 2 SPAZI}IMP
   ARIAMO LE NOTE":PRINT"{ 2 GIU'}
   { 4 SPAZI}LA TASTIERA"                :rem 152
51 SS=36876                              :rem  87
52 DATA131,145,158,161,173,181,189,192
                                         :rem 141
54 FORI=1TO8:READS:POKESS,S:FORD=1TO200:N
   EXTD,I:POKESS,0:GOTO150               :rem 199
55 FORJ=1TO10                            :rem  13
56 PRINT"{CLR}{BLK}":K=INT(RND(1)*2)+1:ON
   KGOSUB20,22:PRINT"[< 22 T>]"          :rem  28
58 N=INT(RND(1)*7)+1:ONKGOTO60,62
                                         :rem 172
60 ONNGOSUB3,4,5,6,8,10,12:GOTO63
                                         :rem 131
62 ONNGOSUB6,8,10,12,14,15,16            :rem  63
63 GOSUB200                              :rem 123
64 POKE7966+CC,42:CS=38664+CC:FORV=0TO2:F
   ORD=1TO20:POKECS+22*V,7:POKECS+22*V,2:
   NEXT:NEXT                             :rem 225
66 PRINT"{GIU'}"                         :rem  78
68 PRINT"{BLU}INDICA LA NOTA{ 5 SPAZI}
   { 5 SIN}";:INPUTA$                    :rem  64
70 IFA$=AA$THEN79                        :rem 221
72 POKE36877,128:FORD=1TO400:NEXT:POKE368
   77,0:PRINT"{ 2 SU}":GOTO68            :rem 211
79 FORT=1TO3:POKE7922+T*22+CC,C(T):NEXT
                                         :rem 119
80 POKESS,S:FORD=1TO400:NEXTD:POKESS,0:NE
   XT                                    :rem 103
90 PRINT"{CLR}{ 3 GIU'}{GRN}VUOI RIPROVAR
   E":PRINT"(S/N)"                       :rem  71
92 INPUTQ$:IFQ$="S"THEN55                :rem  27
94 PRINT"{CLR}{BLU}":END                 :rem   1
104 FORI=1TO7:PRINT"{ 2 SPAZI}{RVS} {OFF}
    {RVS} {OFF} {RVS} {OFF} B {RVS}
    {OFF} {RVS} {OFF} B {RVS} {OFF} {RVS}
    {OFF} {RVS} ":NEXT:RETURN            :rem 189
106 FORI=1TO3:PRINT"{ 2 SPAZI}B B B B B
    B{SPAZI}B{SPAZI}B{SPAZI}B{SPAZI}B":NE
    XT:PRINT"[< 22 Y>]":RETURN           :rem 130
108 PRINT"{ 2 GIU'}{GRN}PREMI RETURN";
                                         :rem  66
109 GETA$:IFA$=""THEN109                 :rem  87
110 IFASC(A$)<>13THEN109                 :rem  85
111 PRINT"{CLR}{BLU}":RETURN             :rem  48
150 PRINT"{CLR}{GIU'}{BLU}LA TASTIERA DI
    UN PIA-NO O DI UN ORGANO HA":rem 182
151 PRINT"GRUPPI DI DUE E TRE"           :rem 218
152 PRINT"TASTI NERI.{ 2 GIU'}{BLK}":GOSU
    B104:GOSUB106:GOSUB108               :rem 239
154 PRINT"{GIU'}OSSERVATE UNA SERIE DI":P
    RINT"DUE TASTI {BLK}NERI{BLU}:"
                                         :rem 166
155 PRINT"{GIU'}I NOMI DEI TASTI SONO DO,
    RE E MI."                            :rem 238
156 PRINT"{BLK}{GIU'}":FORI=1TO7:PRINT"
    { 4 SPAZI}[<H>]{ 2 SPAZI}{RVS}
    { 2 SPAZI}{OFF}{ 2 SPAZI}{RVS}
    { 2 SPAZI}{OFF}{ 3 SPAZI}[<H>]":NEXT
                                         :rem 244
158 PRINT"{ 4 SPAZI}[<H>]{ 3 SPAZI}[<H>]
    { 3 SPAZI}[<H>]{ 3 SPAZI}[<H>]
    { 9 SPAZI}[<H>]{RED}DO{BLK} [<H>]
    {RED}RE{BLK} [<H>]{RED}MI{BLK} [<H>]"
                                         :rem 211
159 PRINT"{ 4 SPAZI}[<H>]{ 3 SPAZI}[<H>]
    { 3 SPAZI}[<H>]{ 3 SPAZI}[<H>]"
                                         :rem  64
160 PRINT" [< 19 Y>]":GOSUB108           :rem  80
164 PRINT"I NOMI DEI TASTI SONO QUELLI DE
    LLE NOTE COR-RISPONDENTI";           :rem 129
165 PRINT", DA FA A{ 2 SPAZI}SI.
    { 3 GIU'}{BLK}"                      :rem 115
166 GOSUB104:PRINT"{ 2 SPAZI}B{SPAZI}B
    {SPAZI}B{SPAZI}B{SPAZI}B{SPAZI}B
    {SPAZI}B{SPAZI}B{SPAZI}B{SPAZI}B
    { 2 SPAZI}{RED}F{BLK}B{RED}S{BLK}B
    {RED}L{BLK}B{RED}S{BLK}B{PUR}D{BLK}B
    {PUR}R{BLK}B{PUR}M{BLK}B{RED}F{BLK}B
    {RED}S{BLK}B{RED}L{BLK}B{RED}S{BLK}";
                                         :rem 247
167 PRINT" {RED}A{BLK}B{RED}O{BLK}B{RED}A
    {BLK}B{RED}I{BLK}B{PUR}O{BLK}B{PUR}E
    {BLK}B{PUR}I{BLK}B{RED}A{BLK}B{RED}O
    {BLK}B{RED}A{BLK}B{RED}I{BLK}";
                                         :rem  51
168 PRINT"{ 2 SPAZI}B{RED}L{BLK}B{SPAZI}
    B{SPAZI}B{SPAZI}B{SPAZI}B{SPAZI}B
    {SPAZI}B{RED}L{BLK}B{SPAZI}B{SPAZI}
    [< 22 Y>]":GOSUB108:GOTO55           :rem 222
170 END                                  :rem 111
200 CC$=LEFT$(AA$+"{ 2 SPAZI}",3):C$(1)=L
    EFT$(CC$,1):C$(2)=MID$(CC$,2,1)
                                         :rem 169
210 C$(3)=RIGHT$(CC$,1):FORT=1TO3:C(T)=AS
    C(C$(T))-64:NEXT                     :rem  17
220 IFC(3)<>12THENC(3)=32                :rem  95
230 RETURN                               :rem 117
```

Word Match (un gioco di memoria per VIC e C64)

di Andy Vanduyne
trad. e adatt. U. Barzaghi

Word Match, gioco educativo per uno o due giocatori, è stato originariamente ideato per il VIC 20 privo di espansione di memoria. È stata aggiunta in seguito la versione per il Commodore 64.

Vagamente ispirato ad un vecchio show televisivo americano ("Concentration"), "Word Match" è studiato per divertire e mettere alla prova le capacità mnemoniche di uno o due giocatori. Lo scopo è di trovare e far coincidere coppie di parole nascoste dietro file di caselle colorate.
È facile imparare a giocare a "Word Match". I giocatori, a turno, scelgono delle caselle, che scompaiono rivelando le parole che nascondevano. Se le parole non vengono accoppiate, ciò significa che la mossa tocca all'altro giocatore. I giocatori che abbinano una coppia di parole, vincono un altro turno, e le caselle accoppiate esattamente cambiano il proprio colore in quello del giocatore. Per vincere la partita, un giocatore deve accoppiare esattamente un numero di coppie di parole nascoste superiore a quello dell'avversario. Detto per inciso, l'avversario, può essere sia un'altra persona che lo stesso calcolatore.
Word Match è ideale per alunni delle scuole elementari, poiché tutte le parole hanno una lunghezza di tre lettere. In totale, nel programma sono incluse 28 parole. (Parole di maggior lunghezza non troverebbero posto nello schermo del VIC, mentre il video a 40 colonne del C 64 consentirebbe di utilizzare anche parole più lunghe).
Potete, personalizzare il programma con un vostro elenco di parole, sostituendo le righe 32 e 33 per la versione VIC e le righe 180, 185 e 190 per il C 64). Ventotto parole dovrebbero rappresentare un limite sufficientemente affidabile. Se desiderate cambiare questo valore, dovete sostituirlo in ogni punto in cui il numero 28 ricorre, alla righe 30, 35, 36 e 38, per la versione VIC, e 170, 200, 210 e 230 per il C 64, con il numero di parole che volete utilizzare.
Le parole inserite nella versione standard sono state scelte per bambini che hanno appena iniziato a leggere.

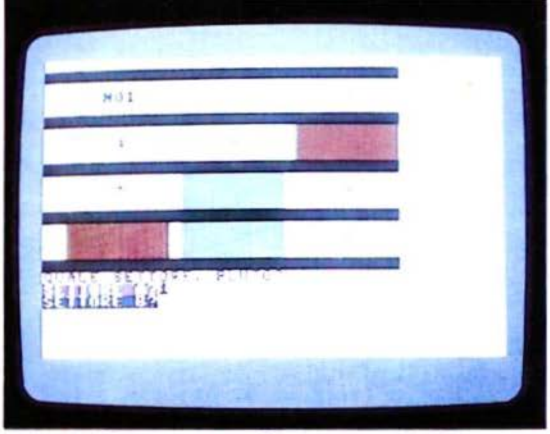
Come si gioca a Word Match

Quando mandate in esecuzione Word Match, il calcolatore chiede, inizialmente, i nomi dei giocatori. Dopo aver digitato il secondo nome da tastiera, il video viene cancellato e, per alcuni momenti, sembra non succedere nulla. Non allarmatevi: il programma richiede del tempo per scegliere le parole da mostrare sullo schermo.

Nonostante il fatto che Word Match sia stato inizialmente ideato per due giocatori, una persona può scegliere di confrontarsi con il calcolatore battendo "VIC" come nome di uno dei due giocatori all'inizio del programma (o "64" con la versione C 64). Il calcolatore, tuttavia, non è così brillante come si potrebbe supporre. Sceglie le caselle completamente a caso. Un bambino può quindi divertirsi senza essere scoraggiato da un avversario imbattibile.
Un interessante esperimento, sia detto per inciso, è rappresentato dal fatto di inserire il nome del calcolatore per entrambi i giocatori e, quindi, star a vedere la macchina che gioca con se stessa. Generalmente il risultato che si ottiene è un errore di "traboccamento" della memoria dopo parecchi turni, ma, a volte, il calcolatore riesce effettivamente a battere se stesso!
Non è permesso utilizzare il tasto di DELETE rispondendo alle domande di scelta del settore. Tuttavia, il programma non accetterà risposte superiori a 12. Se commettete un errore rispondendo alla domanda, limitatevi ad aggiungere qualche cifra in modo da rendere la risposta inaccettabile perché troppo alta. Inoltre, il programma, non accetterà numeri corrispondenti a caselle già accoppiate e, quindi, scoperte.

Ecco un breve riassunto sul funzionamento della versione di Word Match per il VIC.

Linee	Commento
1-13	Dà inizio al programma e acquisisce i nomi dei giocatori.
30	Dimensionamento di alcuni vettori.
32-33	Memorizza le parole.
35-47	Sceglie e registra sei parole.
50-81	Prepara opportunamente lo schermo e memorizza i valori numerici corrispondenti a ciascun settore per reperirne successivamente la posizione.
100-130	Acquisisce la risposta.
132-140	Scopre le parole nascoste.
150-162	Segnala un accoppiamento errato.
200-238	Segnala un accoppiamento esatto, elimina le caselle già accoppiate, e controlla se tutti gli accoppiamenti sono stati effettuati.
300-314	Dichiara la partita conclusa, il vincitore, e chiede se si desidera dare il via ad un'altra partita.
1000	Sottoprogramma che permette al calcolatore di giocare.



Word Match versione per C 64

```
100 POKE53281,1:PRINT"{CLR}"SPC(250)"
    {BLK}{ 2 SPAZI}WORD MATCH " :rem     9
110 FORP=1TO2E3:NEXT:PRINTSPC(90)"{BLU}
    { 2 SPAZI}PREMI QUALSIASI TASTO..."
                                :rem     5
120 GETA$:IFA$=""THEN120        :rem    73
130 DIMN$(2):PRINT"{BLU}{CLR}I NOMI DEGLI
     ISCRITTI,PREGO!"           :rem    54
140 PRINT"{HOME}{ 15 GIU'}SE VUOI GIOCARE
     CONTRO IL COMPUTER,"       :rem    82
145 PRINT"DIGITA '64' COME GIOCATORE"
                                :rem   150
150 PRINT"{HOME}{GIU'}":FORN=1TO2:PRINT"
?   {GIU'}GIOCATORE"N;:INPUTN$(N):NEXT:PR
    INT"{CLR}"                  :rem   113
160 O=54272                     :rem    39
161 S=54272:FORE=STOS+28:POKEE,0:NEXT
                                :rem    48
162 POKE54296, 15 :POKE54277, 66 :POKE542
    78, 210                     :rem   160
163 POKE 54276, 17 :POKE 54272, 63
                                :rem   103
164 J=20:FORT=1TO 200:POKE54273,J:J=RND(0
    )*150+1:NEXT:POKE54276,16   :rem    63
165 FORT=1TO200:NEXT            :rem   243
170 DIMW$(12),R%(28),W1$(6),C%(2),SQ%(12)
    ,BK%(2)                     :rem   127
180 DATABUE,UVA,REO,LUI,LEI,VOI,NOI,UNO,D
    UE,TRE,SEI,MAI,ORA,NEO,ZIO,ZIA,BLU,VI
    A                           :rem   181
185 DATAPIA,REA                 :rem   150
190 DATAAIA,CHI,CHE,SUO,SUD,EST,PIO,SUA
                                :rem   247
200 FORN=1TO28                  :rem    66
210 Z=INT(RND(1)*28)+1:IFR%(Z)<>0THEN210
                                :rem    19
220 R%(Z)=N:NEXT                :rem   186
230 FORN=1TO6:FORQ=1TOR%(N):READW$:NEXT:W
    1$(N)=W$:RESTORE:NEXT:FORQ=1TO28:READ
    W$:NEXT                     :rem    63
240 C%(1)=2:C%(2)=5:BK%(1)=172:BK%(2)=236
                                :rem    18
250 FORN=1TO12:R%(N)=0:NEXT     :rem   118
260 FORN=1TO11STEP2             :rem   174
270 Z=INT(RND(1)*12)+1:Y=INT(RND(1)*12)+1
                                :rem    22
280 IFR%(Z)<>0ORR%(Y)<>0ORZ=YTHEN270
                                :rem   186
290 R%(Z)=N:R%(Y)=N+1           :rem   138
300 NEXT                        :rem   210
310 FORN=1TO12:R%(N)=INT((R%(N)-1)/2)+1:W
    $(N)=W1$(R%(N)):NEXT        :rem   165
320 FORN=1TO4                   :rem    15
330 PRINT"{BLK}{RVS}{ 31 SPAZI}":rem     9
```

L O A D

Word Match (un gioco di memoria per VIC e C64)

```
340 FORZ=1TO3                           :rem    28
350 PRINT"{RVS}{OFF}{ 9 SPAZI}{RVS}{OFF}
    { 9 SPAZI}{RVS}{OFF}{ 9 SPAZI}{RVS}
    {OFF}"                              :rem   249
360 NEXTZ,N                             :rem   172
370 PRINT"{RVS}{ 31 SPAZI}"             :rem   125
380 PRINT"{HOME}{ 2 GIU'}{PUR}"         :rem    61
390 FORN=1TO9STEP3                      :rem   138
400 PRINTTAB(5)NSPC(7)N+1SPC(7)N+2
                                        :rem   253
410 PRINT"{ 2 GIU'}":NEXT               :rem     1
420 PRINT"{ 6 DES}10{ 8 DES}11{ 8 DES}12"
                                        :rem    11
430 D$="{HOME}{ 18 GIU'}"               :rem   197
440 SP$="{ 39 SPAZI}"                   :rem   224
450 PRINTD$SP$SP$                       :rem    28
460 FORN=1TO12:READSQ%(N):NEXT          :rem   122
470 DATA1108,1118,1128,1268,1278,1288,142
    8,1438,1448,1588,1598,1608          :rem    92
480 REM GIOCO                           :rem   241
490 X=1                                 :rem    99
500 PRINTD$"{RED}QUALE SETTORE, "N$(X)"?"
                                        :rem    82
510 POKEBK,BK%(X)                       :rem   217
520 IFN$(X)="64"THENGOSUB1070           :rem   163
530 PRINTD$"{GIU'}"SP$D$"{GIU'}{PUR}{RVS}
    SETTORE A?{BLK}{OFF}";              :rem    23
540 GETA$:IFA$=CHR$(13)THENI=VAL(AN$):AN$
    ="":GOSUB1060:ON(I>12)+2GOTO530,570
                                        :rem   146
550 IFA$<"0"ORA$>"9"THEN540             :rem   199
560 PRINTA$;:AN$=AN$+A$:GOTO540         :rem    65
570 ON(R%(I)=0)+2GOTO530,650            :rem   253
580 PRINTD$"{ 2 GIU'}"SP$D$"{ 2 GIU'}
    {BLU}{RVS}SETTORE B?{BLK}{OFF}";:
                                        :rem   252
590 GETA$:IFA$=CHR$(13)THENJ=VAL(AN$):AN$
    ="":GOSUB1060:ON(J>12)+2GOTO580,620
                                        :rem   154
600 IFA$<"0"ORA$>"9"THEN590             :rem   200
610 PRINTA$;:AN$=AN$+A$:GOTO590         :rem    66
620 ON(R%(J)=0)+2GOTO580,630            :rem   253
630 IFI=JTHEN580                        :rem   196
640 PRINTD$SP$SP$SP$SP$;:GOTO680:rem   247
650 FORN=1TO3                           :rem    20
660 POKESQ%(I)+N+40+O,6:POKESQ%(I)+N+40,A
    SC(MID$(W$(I),N,1))-64:NEXT :rem   133
670 ON(N$(X)="64")+2GOTO1090,580:rem   186
680 FORN=1TO3                           :rem    23
690 POKESQ%(J)+N+40+O,6:POKESQ%(J)+N+40,A
    SC(MID$(W$(J),N,1))-64              :rem    18
700 NEXT                                :rem   214
710 IFR%(I)=R%(J)THEN810                :rem    79
720 PRINTD$SP$D$TAB(12)"{RED}{RVS}SBAGLIA
    TO-{OFF}"                           :rem    15
730 REM SBAGLIATO                       :rem    20
731 S=54272:FORE=STOS+28:POKEE,0:NEXT
                                        :rem    51
732 POKE54296, 15 :POKE54277, 51 :POKE542
    78, 164                             :rem   165
```

```
733 POKE54275, 8 :POKE54274, 100:rem   99
734 POKE 54273, 2 :POKE54272, 135 :POKE54
    276, 65                       :rem  106
735 FORT=1TO 200 :NEXT:POKE54276, 64
                                  :rem  253
740 FORP=1TO2E3:NEXT:PRINTD$SP$ :rem  252
750 I$=STR$(I):I$=RIGHT$(I$,LEN(I$)-1):J$
    =STR$(J):J$=RIGHT$(J$,LEN(J$)-1)
                                  :rem  115
760 POKESQ%(I)+41,32:POKESQ%(J)+41,32:POK
    ESQ%(I)+43,32:POKESQ%(J)+43,32
                                  :rem   29
770 FORN=1TOLEN(I$):POKESQ%(I)+41+N,ASC(M
    ID$(I$,N,1)):POKESQ%(I)+41+N+O,4:NEXT
                                  :rem  101
780 FORN=1TOLEN(J$):POKESQ%(J)+41+N,ASC(M
    ID$(J$,N,1)):POKESQ%(J)+41+N+O,4:NEXT
                                  :rem  106
790 IFX=1THENX=2:GOTO500          :rem  243
800 X=1:GOTO500                   :rem  102
810 REM ESATTO                    :rem   77
820 PRINTD$SP$SP$D$"{ 12 SPAZI}{BLK}{PUR}
    {RVS}ESATTO!!!{OFF}"          :rem  204
830 REM MATCH!!                   :rem   46
831 S=54272:FORE=STOS+28:POKEE,0:NEXT
                                  :rem   52
832 POKE54296, 10 :POKE54277, 35 :POKE542
    78,163                        :rem  162
833 POKE 54273, 33 :POKE54272, 135 :POKE5
    4276,17                       :rem  155
834 FORT=1TO 100 :NEXT            :rem  245
835 POKE 54273, 42 :POKE54272, 62  :POKE5
    4276,17                       :rem  108
836 FORT=1TO 100 :NEXT            :rem  247
837 POKE 54273,50  :POKE54272, 60  :POKE5
    4276,17                       :rem  107
838 FORT=1TO 600 :NEXT:POKE54276, 16
                                  :rem    2
840 IFX=1THENS1=S1+1              :rem  193
850 IFX=2THENS2=S2+1              :rem  197
860 FORP=1TO2E3:NEXT              :rem    9
870 Q=SQ%(I):R=SQ%(J)             :rem  189
880 FORN=1TO3                     :rem   25
890 FORW=Q-2TOQ+6:POKEW+O,C%(X):POKEW,160
    :NEXT:Q=Q+40:NEXT             :rem  181
900 FORN=1TO3                     :rem   18
910 FORW=R-2TOR+6:POKEW+O,C%(X):POKEW,160
    :NEXT:R=R+40:NEXT             :rem  178
920 PRINTD$SP$                    :rem   87
930 CR=CR+1:IFCR=6THEN960         :rem  254
940 R%(I)=0:R%(J)=0               :rem  212
950 GOTO500                       :rem  108
960 FORN=1TO5:FORZ=250TO150STEP-2:POKES,Z
    :NEXTZ,N:POKES,0              :rem  228
970 PRINTD$"IL GIOCO E' FINITO- :rem   31
980 FORP=1TO2E3:NEXT              :rem   12
990 IFS1>S2THENPRINTD$SP$D$N$(1)" HA VINT
    O!!!"                         :rem  127
1000 IFS2>S1THENPRINTD$SP$D$N$(2)" HA VIN
     TO!!!"                       :rem  159
1010 IFS2=S1THENPRINTD$SP$D$"PARITA'!!!"
                                  :rem  121
1020 FORP=1TO2E3:NEXT:PRINTD$"{GIU'}ANCOR
     A (S/N)?"                    :rem  198
1030 GETA$:IFA$="N"THENPRINT"{CLR}{BLU}":
     POKEBK,27:END                :rem   84
1040 IFA$="S"THENRUN130           :rem   69
1050 GOTO1030                     :rem  195
1060 REM SUONO INIZIALE           :rem  148
1061 S=54272:FORE=STOS+28:POKEE,0:NEXT
                                  :rem   96
```

L O A D

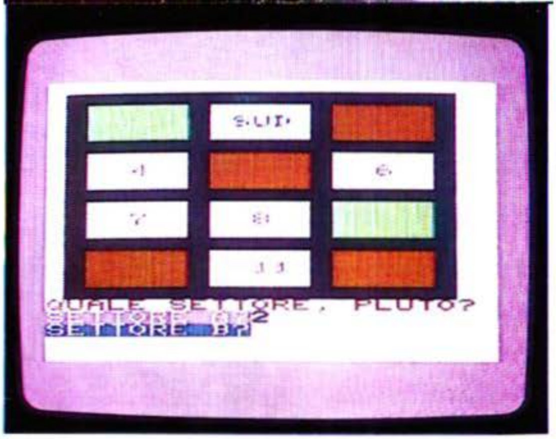
Word Match (un gioco di memoria per VIC e C64)

```
1062 POKE54296, 15 :POKE54277, 51 :POKE54
     278, 246                     :rem  211
1063 POKE 54276,17  :POKE 54273,1  :POKE5
     4272, 135                    :rem  146
1064 FORT=1TO200STEP16  : POKE54273,T/2+5
     0:NEXT:POKE54276,16:FORT=1TO 500 :NE
     XT                           :rem   60
1065 RETURN                       :rem  172
1070 REM GIOCA IL C-64            :rem  126
1080 I=INT(RND(1)*12)+1:ON(R%(I)=0)+2GOTO
     1080,650                     :rem  166
1090 J=INT(RND(1)*12)+1:IFJ=ITHEN1090
                                  :rem  110
1100 IFR%(J)=0THEN1090            :rem  201
1110 PRINTD$SP$D$"64 SCEGLIE"I"E"J"
     { 2 SIN}."                   :rem   89
1120 FORP=1TO2E3:NEXT:GOTO680     :rem   64
```

Word Match versione per VIC 20

```
1 REM WORD MATCH                  :rem  190
2 PRINT"{CLR}"SPC(247)"{PUR}WORD MATCH"
                                  :rem  243
3 FORP=1TO2E3:NEXT:PRINTSPC(66)"{BLU}PREM
  I QUALSIASI TASTO"              :rem   31
4 GETA$:IFA$=""THEN4              :rem  139
10 DIMN$(2):PRINT"{BLU}{CLR}I NOMI DEGLI
   ISCRITTI,PREGO !"              :rem    3
11 PRINT"{HOME}{ 19 GIU'}SE VUOI GIOCARE
   CONTROIL COMPUTER";            :rem  114
12 PRINT",DIGITA{ 4 SPAZI}'VIC' COME GIOC
   ATORE."                        :rem   49
13 PRINT"{HOME}{GIU'}":FORN=1TO2:PRINT"
   {GIU'}GIOCATORE"N;:INPUTN$(N):NEXT:PRI
   NT"{CLR}"                      :rem   63
20 S=36876:V=S+2:BK=V+1:POKES,0:POKEV,15:
   POKEBK,156:O=30720             :rem   28
30 DIMW$(12),R%(28),W1$(6),C%(2),SQ%(12),
   BK%(2)                         :rem   74
32 DATABUE,UVA,REO,LUI,LEI,VOI,NOI,UNO,DU
   E,TRE,SEI,MAI,ORA,NEO,ZIO,ZIA,BLU,VIA,
   PIA,REA                        :rem  139
33 DATAAIA,CHI,CHE,SUO,SUD,EST,PIO,SUA
                                  :rem  195
35 FORN=1TO28                     :rem   24
36 Z=INT(RND(1)*28)+1:IFR%(Z)<>0THEN36
                                  :rem  191
37 R%(Z)=N:NEXT                   :rem  144
38 FORN=1TO6:FORQ=1TOR%(N):READW$:NEXT:W1
   $(N)=W$:RESTORE:NEXT:FORQ=1TO28:READW$
   :NEXT                          :rem   21
39 C%(1)=2:C%(2)=5:BK%(1)=172:BK%(2)=236
                                  :rem  232
40 FORN=1TO12:R%(N)=0:NEXT        :rem   67
41 FORN=1TO11STEP2                :rem  123
42 Z=INT(RND(1)*12)+1:Y=INT(RND(1)*12)+1
                                  :rem  227
```

```
43 IFR%(Z)<>0ORR%(Y)<>0ORZ=YTHEN42
                                    :rem  84
44 R%(Z)=N:R%(Y)=N+1                :rem  87
45 NEXT                             :rem 168
47 FORN=1TO12:R%(N)=INT((R%(N)-1)/2)+1:W$
   (N)=W1$(R%(N)):NEXT              :rem 124
50 FORN=1TO4                        :rem 223
51 PRINT"{BLK} {RVS}{ 19 SPAZI}":rem 217
52 FORZ=1TO3                        :rem 236
53 PRINT" {RVS} {OFF}{ 5 SPAZI}{RVS}
   {OFF}{ 5 SPAZI}{RVS} {OFF}{ 5 SPAZI}
   {RVS} {OFF}"                     :rem 201
54 NEXTZ,N                          :rem 124
55 PRINT" {RVS}{ 19 SPAZI}"         :rem  77
56 PRINT"{HOME}{ 2 GIU'}{PUR}"      :rem  13
60 FORN=1TO9STEP3                   :rem  84
62 PRINTTAB(3)NSPC(3)N+1SPC(3)N+2
                                    :rem 199
64 PRINT"{ 2 GIU'}":NEXT            :rem 214
66 PRINT"{ 3 DES}10{ 4 DES}11{ 4 DES}12"
                                    :rem 162
70 D$="{HOME}{ 18 GIU'}"            :rem 149
71 SP$="{ 20 SPAZI}"                :rem 176
74 PRINTD$SP$SP$                    :rem 238
80 FORN=1TO12:READSQ%(N):NEXT       :rem  72
81 DATA7726,7732,7738,7814,7820,7826,7902
   ,7908,7914,7990,7996,8002        :rem 102
100 REM GIOCO                       :rem 230
105 X=1                             :rem  92
110 PRINTD$"{RED}QUALE SETTORE, "N$(X)"?"
                                    :rem  79
115 POKEBK,BK%(X)                   :rem 218
117 IFN$(X)="VIC"THENGOSUB1000      :rem  22
120 PRINTD$"{GIU'}"SP$D$"{GIU'}{PUR}{RVS}
    SETTORE A?{BLK}{OFF}";          :rem  18
121 GETA$:IFA$=CHR$(13)THENI=VAL(AN$):AN$
    ="":GOSUB400:ON(I>12)+2GOTO120,124
                                    :rem  80
122 IFA$<"0"ORA$>"9"THEN121         :rem 189
123 PRINTA$;:AN$=AN$+A$:GOTO121     :rem  55
124 ON(R%(I)=0)+2GOTO120,132        :rem 238
125 PRINTD$"{ 2 GIU'}"SP$D$"{ 2 GIU'}
    {BLU}{RVS}SETTORE B?{BLK}{OFF}";
                                    :rem 189
126 GETA$:IFA$=CHR$(13)THENJ=VAL(AN$):AN$
    ="":GOSUB400:ON(J>12)+2GOTO125,129
                                    :rem  97
127 IFA$<"0"ORA$>"9"THEN126         :rem 199
128 PRINTA$;:AN$=AN$+A$:GOTO126     :rem  65
129 ON(R%(J)=0)+2GOTO125,130        :rem 247
130 IFI=JTHEN125                    :rem 186
131 PRINTD$SP$SP$SP$SP$;:GOTO137:rem 239
132 FORN=1TO3                       :rem  15
135 POKESQ%(I)+N+22+O,6:POKESQ%(I)+N+22,A
    SC(MID$(W$(I),N,1))-64:NEXT :rem 130
136 ON(N$(X)="VIC")+2GOTO1040,125
                                    :rem  37
137 FORN=1TO3                       :rem  20
138 POKESQ%(J)+N+22+O,6:POKESQ%(J)+N+22,A
    SC(MID$(W$(J),N,1))-64          :rem  15
139 NEXT                            :rem 220
140 IFR%(I)=R%(J)THEN200            :rem  69
150 PRINTD$SP$D$TAB(6)"{RED}{RVS}SBAGLIAT
    O{OFF}"                         :rem 178
151 FORN=1TO30:POKES,240:POKES,240-3*N:NE
    XT:POKES,0                      :rem 165
152 FORP=1TO2E3:NEXT:PRINTD$SP$ :rem 249
153 I$=STR$(I):I$=RIGHT$(I$,LEN(I$)-1):J$
    =STR$(J):J$=RIGHT$(J$,LEN(J$)-1)
                                    :rem 112
154 POKESQ%(I)+23,32:POKESQ%(J)+23,32:POK
    ESQ%(I)+25,32:POKESQ%(J)+25,32
```




```
                                    :rem  26
155 FORN=1TOLEN(I$):POKESQ%(I)+23+N,ASC(M
    ID$(I$,N,1)):POKESQ%(I)+23+N+O,4:NEXT
                                    :rem  98
156 FORN=1TOLEN(J$):POKESQ%(J)+23+N,ASC(M
    ID$(J$,N,1)):POKESQ%(J)+23+N+O,4:NEXT
                                    :rem 103
160 IFX=1THENX=2:GOTO110            :rem 231
162 X=1:GOTO110                     :rem 100
200 REM RISPOSTA ESATTA             :rem 173
205 PRINTD$SP$SP$D$"{ 6 SPAZI}{BLK}{PUR}
    {RVS}ESATTO!!!!{OFF}"           :rem 234
210 FORN=1TO5:FORZ=200TO240:POKES,Z:NEXTZ
    ,N:POKES,0                      :rem  56
211 IFX=1THENS1=S1+1                :rem 185
212 IFX=2THENS2=S2+1                :rem 189
215 FORP=1TO2E3:NEXT                :rem   3
220 Q=SQ%(I):R=SQ%(J)               :rem 178
225 FORN=1TO3                       :rem  18
230 FORW=QTOQ+4:POKEW+O,C%(X):POKEW,160:N
    EXT:Q=Q+22:NEXT                 :rem  72
231 FORN=1TO3                       :rem  15
232 FORW=RTOR+4:POKEW+O,C%(X):POKEW,160:N
    EXT:R=R+22:NEXT                 :rem  78
235 PRINTD$SP$                      :rem  86
237 CR=CR+1:IFCR=6THEN300           :rem 242
238 R%(I)=0:R%(J)=0                 :rem 212
240 GOTO110                         :rem  97
300 FORN=1TO5:FORZ=250TO150STEP-2:POKES,Z
    :NEXTZ,N:POKES,0                :rem 216
305 PRINTD$"IL GIOCO E' TERMINATO"
                                    :rem 246
306 FORP=1TO2E3:NEXT                :rem   4
307 IFS1>S2THENPRINTD$SP$D$N$(1)" HA VINT
    O!!!{ 9 SPAZI}"                 :rem 119
308 IFS2>S1THENPRINTD$SP$D$N$(2)" HA VINT
    O!!!{ 9 SPAZI}"                 :rem 121
309 IFS2=S1THENPRINTD$SP$D$"PARITA'!!!
    { 11 SPAZI}"                    :rem  83
310 FORP=1TO2E3:NEXT:PRINTD$"{GIU'}ANCORA
    ? (S/N)"                        :rem 151
311 GETA$:IFA$="N"THENPRINT"{CLR}{BLU}":P
    OKEBK,27:END                    :rem  37
312 IFA$="S"THENRUN10               :rem 227
314 GOTO311                         :rem 102
337 CR=CR+1:IFCR=6THEN300           :rem 243
400 POKES,249:FORP=1TO20:NEXT:POKES,0:RET
    URN                             :rem 112
1000 REM GIOCO VIC                  :rem 248
1005 I=INT(RND(1)*12)+1:ON(R%(I)=0)+2GOTO
     1005,132                       :rem 155
1040 J=INT(RND(1)*12)+1:IFJ=ITHEN1040
                                    :rem 100
1050 IFR%(J)=0THEN1040              :rem 200
1060 PRINTD$SP$D$"VIC SCEGLIE"I"E"J"{SIN}
     ."                             :rem  56
1065 FORP=1TO2E3:NEXT:GOTO137       :rem  69
```

STOP-LIST

di Dough Ferguson
trad. e adatt. E. Comini

STOP-LIST è una breve routine che consente di fermare, su video, i listati del VIC 20 e del C 64.



Questa routine, pur essendo estremente breve, è indispensabile a chi programma in BASIC.
Permette di effettuare una pausa o di bloccare il LIST di un programma sul video.
La routine è costituita da un programma scritto in linguaggio macchina che viene caricato in memoria da un programma BASIC.
Il programma "gira" indifferentemente su VIC 20 e C64.
Con i computer Commodore, VIC 20 e C64, non è possibile avere una pausa durante la funzione LIST.
Quando si scrive, ma soprattuto quando si corregge un programma, in particolar modo se non si possiede una stampante, si spreca molto tempo ripetendo in continuazione l'esecuzione LIST del programma.
Tutto ciò per riuscire a fermare il listato (in modo definitivo con il tasto STOP) nel punto voluto.
La routine qui presentata è una elegante soluzione del problema.
Essa si inserisce direttamente nella routine di LIST presente nella ROM del sistema operativo senza interferire con alcunché.

Il programma

Il programma presentato attiva nel VIC 20, o nel C64, una routine di pausa dei listati.
Dopo aver battuto il programma BASIC è necessario provvedere al suo salvataggio prima di dare RUN poiché, dopo il caricamento della routine il linguaggio macchina nelle apposite locazioni di memoria, il programma stesso si distrugge (vedere linea 80).
Per utilizzare la routine è sufficiente premere il tasto SHIFT mentre è in atto il LIST di un programma, in tal modo si ottiene una pausa nella visualizzazione del listato.
Per fermare completamente il LIST si premerà il tasto SHIFT-LOCK, per la ripresa del LIST rilasciare il tasto SHIFT oppurre premere nuovamente il tasto **SHIFT-LOCK.**

Qualche dettaglio tecnico

Per chi desidera saperne di più spieghiamo, qui di seguito, il funzionamento della routine proposta.
Nelle linee 20 e 30 si sposta, all'indietro, di 23 byte il valore relativo al top della memoria, in tal modo la routine resta protetta da "invasioni" di variabili o di programmi troppo lunghi.
Inoltre si provvede al cambio del vettore di salto della routine che gestisce la funzione **LIST** in modo che, prima che questa venga eseguita, il controllo passi attraverso la nuova routine.

La linea 20 controlla anche che il programma di caricamento non venga lanciato più volte.

Le istruzioni della linea 40 provvedono alla sistemazione della routine nell'apposito spazio riservato.

Notare che, quando la routine viene attivata, vuota il buffer della tastiera.

Questo accorgimento, inutile ma non dannoso per il C64, si è reso necessario per il VIC 20 in tal modo è stato possibile fornire una sola versione della routine per le due macchine.

Stop-List versione per VIC 20 e C 64

```
20 L=232:H=PEEK(56)-1:Q=PEEK(775):IFQ<167
   THEN80                            :rem  236
30 POKE55,L:POKE51,L:POKE56,H:POKE52,H:PO
   KE774,L:POKE775,H                 :rem   74
40 FORX=L+H*256TOX+21:READD:POKEX,D:NEXT
                                     :rem   51
50 POKEX,Q                           :rem  105
60 DATA72,152,72,32,159,255,169,1,44,141,
   2,208,246                         :rem  209
70 DATA169,0,133,198,104,168,104,76,26
                                     :rem  136
80 NEW                               :rem   82
```

VIC-CLOCK il VIC 20 e il suo orologio

di **Charles Brannon**
trad. e adatt. **E. Comini**

Il VIC 20 possiede, di serie, un preciso orologio che può essere gestito via software.

Ogni sessantesimo di secondo il chip VIA (Versatile Interface Adapter) del VIC produce un interrupt, ordina cioé al microprocessore di fermarsi. Il micro cessa per un istante di fare ciò di cui si stava occupando, per provvedere alla gestione di una particolare routine detta appunto "routine di interrupt".
Dopo l'esecuzione di questa routine il micro riprende la funzione interrotta in precedenza.
Durante l'interrupt e l'esecuzione della routine relativa, il VIC si preoccupa della propria gestione interna, compie infatti determinate azioni per garantire il corretto funzionamento del sistema.
Tra le azioni svolte in tali brevi attimi si ha la lettura della testiera e la trasformazione del codice di scansione rilevato in codice ASCII, tale valore viene inserito nel buffer relativo alla tastiera (un'area di memoria che serve a conservare i codici relativi agli ultimi tasti premuti).
La routine di interrupt provvede a far lampeggiare il cursore, inoltre tutte le volte che viene lanciata incrementa l'orologio in tempo reale.
L'orologio del VIC utilizza 3 locazioni di memoria: la 160, la 161 e la 162.
Il tempo è memorizzato in sessantesimi di secondo, ed è aggiornato ogni sessantesimo di secondo.
Poiché un byte può contenere un numero il cui valore dev'essere compreso tra 0 e 255 si è reso necessario utilizzare tre byte per un orologio che potesse funzionare per 24 ore. Se proviamo infatti a calcolare il numero 2 elevato a 24 rappresentabile con 3 byte e lo si divide prima per 60 poi per 3600 si otterrà un valore di poco superiore a 77 (e) tale è il numero di ore che l'orologio a 3 byte potrebbe contare prima di tornare nuovamente a zero.
Utilizzando 2 byte l'orologio tornerebbe a zero dopo (2^16)/60)/3600, cioé dopo circa un quarto d'ora.
Ogni volta che la locazione 162 raggiunge il valore 0 (dopo circa 4 secondi) viene incrementata la locazione 161 che a propria volta raggiungendo il valore 0 incrementa di 1 la locazione 160.
Notare che i tre byte sono disposti nel senso contrario rispetto all'ordine convenzionale del 6502 in cui i byte di minor peso (meno significativi) vengono prima di quelli di maggior peso (più significativi).
Il contenuto dell'orologio in tempo reale può essere comodamente letto utilizzando il BASIC senza fare riferimento alle locazioni di memoria.
Ci sono due variabili riservate a tale scopo: **TIME** e **TIME$** che contengono il tempo corrente (0 all'accensione del calcolatore).
La variabile numerica **TIME** contiene il tempo in sessantesimi di secondo ed è l'equivalente di

PEEK(160) + PEEK(161)*256 + PEEK(162)*65536.

Dividendo per 60 la variabile **TIME** si ottiene il tempo in secondi.
Non è possibile assegnare direttamente un valore alla variabile **TIME**, ad esempio TIME = 0, ma è possibile modificare il valore della variabile alfanumerica **TIME$**
TIME$** è una stringa formata da 6 numeri con il classico formato HHMMSS (HH = ore, MM = minuti, SS = secondi), quindi **TIME$ è un orologio capace di 24 ore.
Per posizionare l'orologio è sufficiente battere: **TIME$ = "HHMMSS"**, ad esempio se si desidera posizionare l'orologio sulle 10.30 battere **TIME$ = "103000"** oppure, per le 22.30 battere **TIME$ = "223000"**.
Dopo aver assegnato il valore, l'orologio procede al conteggio in modo automatico.
La variabile **TIME** e **TIME$** possono essere abbreviate in **TI** e **TI$**.

Il programma

Utilizzando il programma proposto è necessario porre attenzione ad alcune cose.
L'Input/Output dei dati da cassetta utilizza il chip VIA per il trasferimento dati tra memoria e cassetta, infatti durante le operazioni di LOAD e di SAVE è facilmente constatabile che non tutte le routine di interrupt funzionano (premendo un tasto, tranne quello relativo allo STOP, non succede alcunché).
Il programma proposto, che visualizza l'orologio sul video, rimane inattivo durante le operazioni di I/O (Input/Output) da cassette per riprendere il corretto funzionamento al termine di tali operazioni.
L'utilizzo è quello di operare con il VIC tenendo sotto controllo l'ora, tenendo presente che durante operazioni di LOAD e SAVE l'orologio smette di funzionare.
Dopo il fatidico RUN il programma propone diverse opzioni selezionabili con i tasti di funzione: partenza, arresto, azzeramento e regolazione dell'ora corrente.
L'orologio viene visualizzato in alto a sinistra e resta in tale posizione anche dopo uno "scroll" oppure dopo un CLEAR SCREEN.
L'orologio visibile sul video è separato da quello in tempo reale contenuto nelle variabili **TIME** e **TIME$** pertanto il loro utilizzo e la loro manipolazione resta possibile all'interno di un programma.
Il programma per l'orologio è stato aggiunto alla routine di interrupt cui si è accennato all'inizio dell'articolo.
Per mezzo dei tasti STOP + RESTORE l'orologio scompare e la routine inserita non viene più utilizzata.
Dopo aver lanciato il programma e posizionato l'orologio dare NEW liberando così la memoria del programma che ha caricato la routine in linguaggio macchina.
Per fermare l'orologio si utilizzi l'istruzione POKE 997,1 e per farlo ripartire POKE 997,0.
Il colore relativo alle cifre dell'orologio si può modificare con POKE 996,x dove "x" è un numero compreso tra 0 e 7.

LOAD

VIC-CLOCK Il VIC 20 e il suo orologio

VIC-Clock versione per VIC 20

```
100 PRINT"{CLR}{RVS}ATTENDI UN ATTIMO"
                              :rem  129
110 FORI=828TO995:READA:POKEI,A:CK=CK+A:N
    EXT                       :rem   25
120 IFCK<>20518THENPRINT"{HOME}ERRORE NEI
     DATI NUMERICI":END       :rem  119
125 SYS828                    :rem   57
130 PRINT"{CLR}{ 4 SPAZI}{ 4 GIU'}SCEGLI:
    {GIU'}"                   :rem   62
140 PRINT"{RVS}{YEL}F1{OFF}:{BLU}FERMA OR
    OLOGIO"                   :rem   77
150 PRINT"{GIU'}{RVS}{RED}F3{OFF}{BLU}:PA
    RTE OROLOGIO"             :rem  240
160 PRINT"{GIU'}{RVS}{PUR}F5{OFF}{BLU}:AZ
    ZERA OROLOGIO"            :rem  196
165 PRINT"{GIU'}{RVS}{GRN}F7{OFF}:{BLU}PO
    SIZIONA OROLOGIO"         :rem   60
170 GETA$:IFA$<CHR$(133)ORA$>CHR$(136)THE
    N170                      :rem   39
180 ONASC(A$)-132GOTO190,240,250,200
                              :rem  156
190 POKE997,1:GOTO170         :rem  218
200 POKE997,1:INPUT"{CLR}{ 2 GIU'}ORE? 00
    { 4 SIN}";H$:IFLEN(H$)<>2THEN200
                              :rem  252
210 INPUT"MINUTI? 00{ 4 SIN}";M$:IFLEN(M$
    )<>2THEN210               :rem  212
220 INPUT"SECONDI? 00{ 4 SIN}";S$:IFLEN(S
    $)<>2THEN220              :rem   17
230 T$=H$+M$+S$+"00":FORI=1TO8:POKE998+I,
    ASC(MID$(T$,I)):NEXT:GOTO130:rem   13
240 POKE997,0:GOTO170         :rem  213
250 SYS851:POKE997,1:GOTO170  :rem  174
828 DATA173,020,003,141,226,003 :rem   27
834 DATA173,021,003,141,227,003 :rem   26
840 DATA120,169,098,141,020,003 :rem   33
846 DATA169,003,141,021,003,169 :rem   39
852 DATA048,162,009,157,230,003 :rem   40
858 DATA202,208,250,142,229,003 :rem   39
864 DATA088,096,173,229,003,208 :rem   60
870 DATA053,162,008,024,189,230 :rem   43
876 DATA003,105,001,141,228,003 :rem   26
882 DATA201,058,208,005,169,048 :rem   51
888 DATA141,228,003,138,041,001 :rem   37
894 DATA240,012,173,228,003,201 :rem   33
900 DATA054,208,005,169,048,141 :rem   41
906 DATA228,003,173,228,003,157 :rem   43
912 DATA230,003,202,240,007,201 :rem   14
918 DATA048,208,249,076,105,003 :rem   51
924 DATA162,008,160,011,173,002 :rem   28
930 DATA144,010,169,000,133,251 :rem   27
936 DATA042,010,133,252,133,254 :rem   33
942 DATA173,005,144,074,074,074 :rem   47
948 DATA005,252,133,252,169,000 :rem   41
954 DATA133,253,165,254,009,148 :rem   54
960 DATA133,254,173,240,003,145 :rem   37
966 DATA253,189,230,003,009,128 :rem   51
972 DATA145,251,138,041,001,240 :rem   34
978 DATA010,136,169,058,145,251 :rem   56
984 DATA173,240,003,145,253,136 :rem   45
990 DATA202,208,225,076,049,234 :rem   50
```

Sul prossimo numero troverete:

- **MLX Uno strepitoso programma di utilità che vi permette di battere in modo rivoluzionario programmi in linguaggio macchina assolutamente a prova d'errore!**

- **DEMONI DI OSIRIS Un bellissimo gioco spaziale totalmente in linguaggio macchina per il VIC ed il C64, con un'ottima grafica ed una elevatissima velocità di gioco.**

- **COLORBOT Tocca a voi difendere la Terra dai terribili robot computerizzati che minacciano l'estinzione del genere umano. Coloratissimo e tutto in linguaggio macchina.**

- **OGNI MESE un diverso spartito musicale per il MUSIC EDITOR pubblicato su questo numero, ed...**

- **INOLTRE troverete tanti altri utili e divertenti programmi per il vostro VIC o C64.**

VIC-sound

Music Editor per Commodore 64

Scopo di questo editor musicale è sia quello di "fare della musica" che di creare una libreria di spartiti musicali.
Sarà infatti possibile, per chi possiede un floppy disk drive, registrare i brani musicali digitati, ed eventualmente utilizzare gli stessi dati per altri programmi.

Per generare una nota utilizzando l'integrato sonoro di cui il C 64 è dotato (SID), è necessario stabilirne la forma d'onda, che determina il timbro del suono, il volume del suono stesso ed i quattro parametri che definiscono l'andamento della nota.
La figura 1 permette di visualizzare questi quattro parametri: l'ATTACK definisce la velocità con cui la nota raggiunge il livello massimo; il DECAY definisce la velocità con cui la nota passa dal livello massimo al livello di SUSTAIN; infine il RELEASE determina la velocità con la quale la nota torna la livello O.
Ogni nota è caratterizzata da una frequenza diversa, e nel passaggio da un'ottava a quella immediatamente superiore tale frequenza raddoppia. Quindi, poiché un'ottava comprende dodici note, si ha che la frequenza di ogni nota è pari a quella della nota precedente moltiplicata per $\sqrt{12}$.
La frequenza fondamentale per la creazione della scala musicale è di 440 Hz e corrisponde al LA.
Un volta dato il RUN, comparirà sul video la maschera riportata in figura 2. Nella metà sinistra dello schermo vanno scritte la durata (DUR.) e le note suonate dalle tre voci (V.1, V.2 e V.3): è possibile registrare fino a 600 note per ciascuna voce.
Nel corso dell'esecuzione, ogni nota appartenente alla stessa voce viene suonata finché non si incontra un'altra nota o la stringa "XXX" che ne determina l'interruzione.
Nella metà destra dello schermo, nel riquadro superiore, oltre al nome del brano, sono presenti i nove campi che servono a settare i registri del SID:

- **DC1** e **DC2**: vengono utilizzati solo nel caso in cui il suono sia generato per mezzo dell'onda quadra, e ne definiscono il "Duty Cycle", ovvero il rapporto fra ON e OFF dell'onda;
- **W**: definisce la forma d'onda. Può assumere quattro valori: 129 per il rumore bianco, 65 per l'onda quadra, 33 per l'onda a dente di sega e 17 per l'onda triangolare;
- **A/D**: i valori dell'Attack e del Decay sono contenuti in un solo byte. L'ATTACK nei quattro bit di sinistra, il Decay nei quattro bit di destra;
- **S/R**: analogamente al campo precedente, anche per SUSTAIN e RELEASE viene utilizzato un solo byte;
- **C1** e **C2**: frequenze di taglio del filtro;
- **FON** determina l'accensione del filtro e l'entità della risonanza;
- **LEV**: stabilisce il volume del suono ed il tipo di filtro.

Ognuno degli ultimi quattro occupa 1 solo registro per le tre voci.
Tutti i campi vengono confermati sia col tasto RETURN che con uno qualsiasi dei tasti cursore.
Nel riquadro inferiore sono richiamate le operazioni eseguite dai tasti funzione:

- **F1**: esegue il brano in memoria;

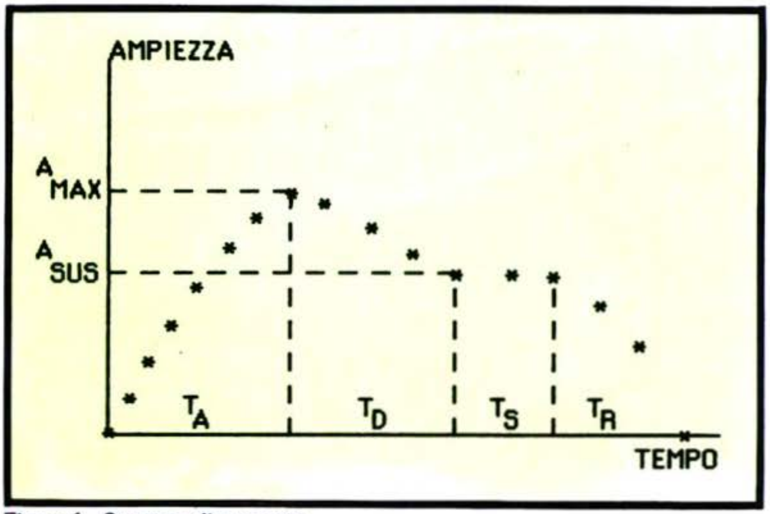


Figura 1 Struttura di una nota.

- **F2**: registra su floppy il contenuto dei registri del SID e le frequenze delle note digitali;
- **F3**: carica il brano dal floppy e lo visualizza sul video;
- **F4**: stampa i valori contenuti nei registri del SID e le note digitate;
- **F5**: rallenta l'esecuzione del brano;
- **F6**: accelera l'esecuzione del brano;
- **F7**: interrompe l'esecuzione del brano;
- **F8**: permette il passaggio del cursore dalla metà destra alla metà sinistra del video, e viceversa;
- **CLR** (SHIFT + CLR/HOME): cancella il brano in memoria. Va utilizzato prima di caricare in memoria un nuovo brano.

Le note vanno scritte usando la notazione anglosassone (ciò allo scopo di risparmiare spazio sul video), e seguite dal numero indicante l'ottava (da 0 a 7). La figura 3 mostra l'equivalenza fra la notazione italiana e quella anglosassone.
Per sveltire la scrittura dello spartito e l'esecuzione del programma vengono utilizzati solo i "diesis" (#) (qualsiasi "bemolle" può essere sostituito dal

DO	= C
DOII(REb)	= CII(Db)
RE	= D
REII(MIb)	= DII(Eb)
MI	= E
FA	= F
FAII(SOLb)	= FII(Gb)
SOL	= G
SOLII(LAb)	= GII(Ab)
LA	= A
LAII(SIb)	= AII(Bb)
SI	= B

Figura 3
Equivalenza fra la notazione italiana e la notazione anglosassone dei nomi delle note.

diesis della nota precedente).
La durata delle note rappresenta solo la durata "relativa" e può essere variata nel corso dell'esecuzione. Come primo spartito viene presentata la 3 "Bourrée in MI()min" di J.S. Bach.

```
     DUR. V.1 V.2 V.3  |BRANO:
                       |  BOURREE
001  02   E 5 G 3      |
002  02   F#5 F#3      |DC1000 000 000
003  04   G 5 E 3      |DC2000 000 000
004  02   F#5 A 3      |W  017 017 017
005  02   E 5          |A/D018 000 128
006  04   D#5 B 3      |S/R128 120 128
007  02   E 5 A 3      |C1 0
008  02   F#5          |C2 000
009  04   B 4 G 3      |FON000
010  02   C#5 F#3      |LEV015
011  02   D#5          |
012  04   E 5 E 3      |F1  :EXECUTE
013  02   D 5 F#3      |F2  :SAVE
014  02   C 5          |F3  :LOAD
015  04   B 4 G 3      |F4  :PRINT
016  02   A 4 A 3      |F5  :SLOW
017  02   G 4          |F6  :FAST
018  04   F#4 B 3      |F7  :STOP
019  02   G 4 A 3      |F8  :SHIFT
020  02   A 4          |CLR:CLEAR
```

Figura 2 Maschera del video.

Linea	Commento
300-570	Routine di gestione del cursore
700	Routine di indirizzamento diretto del cursore
1000-1900	Routine per l'esecuzione del brano
2000-2620	Routine per la registrazione del brano floppy disk
3000-3900	Routine per il caricamento del brano floppy disk
4000-4540	Routine di stampa
10000-10220	Dati per lo scrolling a finestra della parte sinistra dello schermo. Vengono caricati con la linea 60.

```
BRANO: BOURREE9                 N.  DUR.  V.1  V.2  V.3     012  04   E 5  E 3
                                                            013  02   D 5  F#3
                                    001  02   E 5  G 3      014  02   C 5
                                    002  02   F#5  F#3      015  04   B 4  G 3
     DC1: 000  000  000             003  04   G 5  E 3      016  02   A 4  A 3
     DC2: 000  000  000             004  02   F#5  A 3      017  02   G 4
     W  : 017  017  017             005  02   E 5           018  04   F#4  B 3
     A/D: 018  000  128             006  04   D#5  B 3      019  02   G 4  A 3
     S/R: 128  120  128             007  02   E 5  A 3      020  02   A 4
     C1 : 0                         008  02   F#5           021  02   B 4  G 3
     C2 : 000                       009  04   B 4  G 3      022  02   A 4
     FON: 000                       010  02   C#5  F#3      023  02   G 4  B 3
     LEV: 015                       011  02   D#5           024  02   F#4
```

025	02	E 4	E 3	
026	02		F#3	
027	02	E 5	G 3	
028	02	F#5	F#3	
029	04	G 5	E 3	
030	02	F#5	A 3	
031	02	E 5		
032	04	D#5	B 3	
033	02	E 5	A 3	
034	02	F#5		
035	04	B 4	G 3	
036	02	C#5	F#3	
037	02	D#5		
038	04	E 5	E 3	
039	02	D 5	F#3	
040	02	C 5		
041	04	B 4	G 3	
042	02	A 4	C 4	
043	02	G 4		
044	04	F#4	D 4	
045	02		D 4	
046	02			
047	02	G 4		
048	16	G 4	D 4	G 3
049	02	B 4	G 3	
050	02	G 4		
051	04	D 5	F#3	
052	02	A 4	D 4	
053	02	C 5		
054	04	B 4	G 3	
055	02	G 5	B 3	
056	02	D 5		
057	04	E 5	C 4	
058	02	B 4	G#3	
059	02	D 5		
060	04	C 5	A 3	
061	02	B 4	D 4	
062	02	A 4		
063	04	G#4	E 4	
064	02	A 4	A 3	
065	02	B 4		
066	04	C 5	E 4	
067	02	B 4	E 3	
068	02	A 4		
069	02	A 4	A 3	
070	02		B 3	
071	02		A 3	
072	02		G 3	
073	04		F#3	
074	02	D 5	D 4	
075	02	A 4		
076	04	B 4	G 3	
077	02	G 5	B 3	
078	02	D 5		
079	04	E 5	C 4	
080	02	B 4	G#3	
081	02	D 5		
082	04	C 5	A 3	
083	02	A 5	C#4	
084	02	E 5		
085	04	F#5	D 4	
086	02	C#5	A#3	
087	02	E 5		
088	04	D 5	B 3	
089	02	C#5	E 4	
090	02	B 4		
091	04	A#4	F#4	
092	02		F#3	
093	02	B 4		
094	02	B 4	B 3	
095	02		A#3	
096	02		B 3	
097	02		C#4	
098	04		D#4	
099	02	B 5	B 3	
100	02	F#5		
101	04	G#5	E 4	
102	02	F#5	D 4	
103	02	E 5		
104	04	A 5	C#4	
105	02	E 5	A 3	
106	02	G 5		
107	04	F#5	D 4	
108	02	E 5	C 4	
109	02	D 5		
110	04	G 5	B 3	
111	02	D 5	G 3	
112	02	F 5		
113	04	E 5	C 4	
114	02	A 5	B 3	
115	02	E 5		
116	04	F#5	A 3	
117	02	C#5	F#3	
118	02	E 5		
119	04	D#5	B 3	
120	02	B 4		
121	02		C 4	
122	02		B 3	
123	02		A 3	
124	02	E 5	G#3	
125	02	B 4		
126	02	C 5	A 3	
127	02		E 4	
128	02	D 5	F#4	
129	02	A 4		
130	02	B 4	G 3	
131	02		D#4	
132	02	C 5	E 4	
133	02	G 4		
134	02	A 4	F#3	
135	02		C#4	
136	02	B 4	D#4	
137	02	F#4		
138	04	G 4	E 3	
139	02	F#4	A 3	
140	02	E 4		
141	04	D#4	B 3	
142	02	E 4	A 3	
143	02	F#4		
144	04	G 4	B 3	
145	02	F#4		
146	02	E 4		
147	16	E 4	E 3	

Spartito della "Bourrée in MI Min" di J.S. Bach

```
1 REM ***********************       :rem 177
2 REM *      MUSIC EDITOR     *     :rem 178
3 REM *          PER          *     :rem  82
4 REM *     COMMODORE 64      *     :rem 123
5 REM ***********************       :rem 181
10 GOSUB8000:GOSUB760:SZ=400:L0=54272:K0=
   .05961:F0=440.0:I$="{ 10 DES}"
                                    :rem   1
20 J$="{ 10 SPAZI}":DIMF(SZ,4),L%(SZ,3),H
   %(SZ,3),SP$(10,3)                :rem 253
30 FORV=0TO3:READXV(V),LL(V):NEXT:READN$,
   SP$(0,1)                         :rem  11
40 FORN=1TO5:FORV=1TO3:READSP$(N,V):NEXTV
   ,N:FORN=6TO9:READSP$(N,1):NEXT
                                    :rem 113
50 I=0                              :rem  27
60 READJ:IFJ<256THENPOKE49152+I,J:I=I+1:G
   OTO60                            :rem  67
70 Y0=1:N0=1:V1=1:SH=1:GOSUB770:GOTO600
                                    :rem  87
100 YM=SZ:YN=1                      :rem 112
110 N=N0:V=V0:GOSUB750              :rem  68
120 XX=XV(V):LL=LL(V):C=1           :rem  14
130 GOSUB300                        :rem 167
150 IFSH=1THEN600                   :rem 243
160 GOSUB210:IFERTHENGOSUB960:GOTO120
                                    :rem  94
170 GOSUB500:N0=N:V0=V              :rem  67
180 IFN0<Y0THENGOSUB800:GOSUB950:rem  10
190 IFN0>Y0+19THENGOSUB900:GOSUB950
                                    :rem 163
200 GOTO110                         :rem  93
210 IFINP$=LEFT$(I$,LL)ANDER=0THENRETURN
                                    :rem  33
215 IFINP$=LEFT$(J$,LL)THENF(N,V)=0:RETUR
    N                               :rem  94
```

```
220 ER=0:IFV>0THEN240                :rem 234
225 FORI=1TOLL:B=ASC(MID$(INP$,I,1)):IFB<
    48ORB>57THEN265                  :rem 139
230 NEXT:F(N,V)=VAL(INP$):RETURN:rem  11
235 IFINP$="***"THENF(N,V)=-1:RETURN
                                     :rem 126
240 I=ASC(LEFT$(INP$,1)):IFI<65ORI>71THEN
    265                              :rem  97
245 D=ASC(MID$(INP$,2,1)):IFD<>32ANDD<>35
    THEN265                          :rem  10
250 J=ASC(RIGHT$(INP$,1))-48:IFJ<0ORJ>7TH
    EN265                            :rem 229
255 FORI=0TO11:II=I*2+1:IFLEFT$(INP$,2)<>
    MID$(N$,II,2)THENNEXT:GOTO265
                                     :rem 239
260 I=I+J*12:F(N,V)=F0*(2↑(1/12))↑(I-57):
    GOSUB1900:RETURN                 :rem  11
265 ER=1:RETURN                      :rem 188
300 GOSUB700:IFVSTHENPRINTINP$;:RETURN
                                     :rem  56
310 INP$=LEFT$(I$,LL)                :rem 173
320 RV=0:GOSUB480                    :rem   0
330 GETC$:IFC$=""THEN330             :rem  83
335 A=ASC(C$):IFA=20ANDC>1THENGOSUB480:PR
    INT"{SIN}";:C=C-1:GOTO320        :rem 103
340 IFC>LLTHEN365                    :rem  12
345 IFSH=1THEN360                    :rem 252
350 IFA=32ORA=35OR(A>47ANDA<58)OR(A>64AND
    A<72)THEN470                     :rem 136
355 GOTO365                          :rem 116
360 IFA=32OR(A>47ANDA<58)OR(A>64ANDA<91)T
    HEN470                           :rem   3
365 GOSUB480:IFA=13ORA=17ORA=29ORA=145ORA
    =157THENRETURN                   :rem 250
370 IFA=133THENGOSUB580:GOSUB1000:GOTO450
                                     :rem  17
375 IFA=137THENGOSUB580:GOSUB2000:GOTO450
                                     :rem  27
380 IFA=134THENGOSUB580:GOSUB3000:GOTO450
                                     :rem  21
385 IFA=138THENGOSUB580:GOSUB4000:GOTO450
                                     :rem  31
390 IFA=140THENGOSUB580:SH=-SH:GOTO450
                                     :rem 112
395 IFA=147THENCLR:RUN               :rem 137
400 GOTO320                          :rem  98
450 IFSH=1THENN=N1:V=V1:RETURN       :rem 216
460 N=N0:V=V0:RETURN                 :rem  16
470 PRINTC$;:INP$=LEFT$(INP$,C-1)+C$+MID$
    (INP$,C+1):C=C+1:GOTO320         :rem   7
480 ML=1024+YY*40+XX+C-1:IFRV=0THENRV=1:P
    OKEML,PEEK(ML)OR128:GOTO490 :rem 247
485 IFRV=1THENRV=0:POKEML,PEEK(ML)AND127
                                     :rem  52
490 POKE54272+ML,14:RETURN           :rem  63
500 IFA=13ORA=29THENV=V+1:IFV>3THENV=0:A=
    17                               :rem 166
510 IFA=157THENV=V-1:IFV<0THENV=0:A=145
                                     :rem 133
520 IFA=17ANDN<YMTHENN=N+1           :rem 115
530 IFA=145ANDN>YNTHENN=N-1          :rem 171
540 IFSH=-1THENRETURN                :rem 109
550 IFV=0THENV=1                     :rem 223
560 IF(N<1ORN>5)ANDV>1THENV=1:IFN<YMTHENN
    =N+1                             :rem 128
570 RETURN                           :rem 124
580 IFSH=1THENN1=N:V1=V:RETURN       :rem 220
585 N0=N:V0=V:RETURN                 :rem  24
600 YM=9:YN=0                        :rem   0
610 N=N1:V=V1:C=1:GOSUB650:GOSUB300
                                     :rem 130
```

VIC sound

Music Editor per Commodore 64

```
620 IFSH=-1THEN100                   :rem  29
630 GOSUB660:IFERTHENGOSUB960:GOTO610
                                     :rem 109
640 GOSUB500:N1=N:V1=V:GOTO610       :rem  81
650 XX=VAL(LEFT$(SP$(N,V),2)):YY=VAL(MID$
    (SP$(N,V),3,2))                  :rem 243
655 LL=VAL(MID$(SP$(N,V),5,2)):RETURN
                                     :rem 185
660 IFINP$=LEFT$(I$,LL)ANDER=0THENRETURN
                                     :rem  42
665 IFINP$=LEFT$(J$,LL)THEN690       :rem  24
670 ER=0:IFN1=0THEN690               :rem  36
675 IFVAL(INP$)>255THEN695           :rem  29
680 FORI=1TOLL:B=ASC(MID$(INP$,I,1)):IFB<
    48ORB>57THEN695                  :rem 151
685 NEXT                             :rem 226
690 GOSUB3900:RETURN                 :rem   5
695 ER=1:RETURN                      :rem 195
700 POKE211,XX:POKE214,YY:SYS58640:RETURN
                                     :rem 110
750 XX=0:YY=N+2-Y0:GOSUB700:PRINT"B"RIGHT
    $("000"+MID$(STR$(N),2),3);:RETURN
                                     :rem 112
760 XX=22:YY=24:GOSUB700:PRINT"{RVS}UN MO
    MENTO PREGO!{OFF}";:RETURN      :rem  50
770 XX=22:YY=24:GOSUB700:PRINT"
    { 17 SPAZI}";:RETURN             :rem  47
800 POKE255,14:SYS49227:Y0=Y0-1:RETURN
                                     :rem  54
900 POKE255,14:SYS49152:Y0=Y0+1:RETURN
                                     :rem  50
950 VS=1:IFF(N,0)THENGOSUB3100       :rem  50
955 VS=0:RETURN                      :rem 211
960 GOSUB700:PRINT RIGHT$(J$,LL):RETURN
                                     :rem 188
1000 GOSUB760:FORN=1TO5:FORV=1TO3:M=L0+(V
     -1)*7+N+1:POKEM,0               :rem   6
1010 GOSUB1200:IFN<>3THENPOKEM,S:GOTO1030
                                     :rem  42
1020 W(V)=S                          :rem  81
1030 NEXTV,N:M=M-5:V=1:FORN=6TO9:GOSUB120
     0:POKEM+N,S:NEXT:GOSUB770       :rem 254
1040 IFK=0THENK=1                    :rem 244
1050 FORN=1TOSZ:IFF(N,0)=0THEN1190
                                     :rem  42
1070 IFF(N,1)>0THENPOKE54272,L%(N,1):POKE
     54273,H%(N,1):POKE54276,W(1)
                                     :rem 123
1075 IFF(N,2)>0THENPOKE54279,L%(N,2):POKE
     54280,H%(N,2):POKE54283,W(2)
                                     :rem 135
1080 IFF(N,3)>0THENPOKE54286,L%(N,3):POKE
     54287,H%(N,3):POKE54290,W(3)
                                     :rem 138
1090 FORDY=1TOF(N,0):FORI=1TOK       :rem 102
1100 GETA$                           :rem   7
1105 IFA$="{F5}"THENGOSUB1500        :rem  56
1110 IFA$="{F6}"THENGOSUB1600        :rem  57
1115 IFA$="{F7}"THEN1190             :rem 191
1120 NEXTI,DY                        :rem  21
1125 FORV=1TO3:IFF(N,V)THENM=L0+(V-1)*7+4
```

```
     :POKEM,W(V)+(W(V)>0)           :rem 103
1130 NEXT                           :rem   4
1135 FORDY=1TOF(N,0):FORI=1TOK:NEXT:NEXT
                                    :rem  88
1140 NEXT                           :rem   5
1190 GOSUB1700:RETURN               :rem  45
1200 S=0:IFLEN(SP$(N,V))>6THENS=VAL(MID$(
     SP$(N,V),7))                   :rem 101
1210 RETURN                         :rem 164
1500 IFK<1E+36THENK=K+1             :rem  68
1510 RETURN                         :rem 167
1600 IFK>1THENK=K-1                 :rem 112
1610 RETURN                         :rem 168
1700 POKEL0+4,0:POKEL0+11,0:POKEL0+18,0:P
     OKEL0+24,0:RETURN              :rem 189
1900 H=INT(F(N,V)/K0)/256:L%(N,V)=256*(H-
     INT(H)):H%(N,V)=INT(H):RETURN
                                    :rem  37
2000 GOSUB2500:IFBR$=""THENRETURN
                                    :rem  26
2010 GOSUB760:OPEN2,8,2,"@0:"+BR$+",S,W"
                                    :rem 174
2020 FORN=1TO5:FORV=1TO3:GOSUB2600:NEXTV,
     N                              :rem 197
2030 V=1:FORN=6TO9:GOSUB2600:NEXT:PRINT#2
     ,K                             :rem 213
2040 FORN=1TOSZ                     :rem 185
2050 PRINT#2,F(N,0):IFF(N,0)=0THEN2070
                                    :rem 133
2060 FORV=1TO3:PRINT#2,F(N,V):NEXTV,N
                                    :rem  65
2070 CLOSE2:GOSUB770:RETURN         :rem 227
2500 BR$=J$:IFLEN(SP$(0,1))>6THENBR$=MID$
     (SP$(0,1),7)                   :rem 183
2510 FORI=10TO1STEP-1:IFMID$(BR$,I,1)="
     {DES}"THENNEXT                 :rem 178
2520 BR$=LEFT$(BR$,I):RETURN        :rem 165
2600 SP$=MID$(SP$(N,V),7):IFSP$=""THENGOS
     UB655:SP$=RIGHT$("000",LL)     :rem 143
2610 IFPRTHENRETURN                 :rem   9
2620 PRINT#2,SP$:RETURN             :rem 185
3000 GOSUB2500:IFBR$=""THENRETURN
                                    :rem  27
3010 GOSUB760:OPEN2,8,2,BR$+",S,R"
                                    :rem 145
3020 FORN=1TO5:FORV=1TO3:GOSUB3600:NEXTV,
     N                              :rem 199
3030 V=1:FORN=6TO9:GOSUB3600:NEXT:INPUT#2
     ,K                             :rem 218
3040 FORN=1TOSZ                     :rem 186
3050 INPUT#2,F(N,0):IFF(N,0)=0THEN3070
                                    :rem 138
3060 FORV=1TO3:INPUT#2,F(N,V.:GOSUB1900:N
     EXTV,N                         :rem 201
3070 CLOSE2:VS=1:GOSUB770           :rem  27
3075 FORN=1TO5:FORV=1TO3:GOSUB3200:NEXTV,
     N:V=1:FORN=6TO9:GOSUB3200:NEXT
                                    :rem 129
3080 FORN=1TOSZ:IFF(N,0)=0THEN3095
                                    :rem  53
3085 IFN>20THENGOSUB900             :rem 149
3090 GOSUB3100:NEXT                 :rem 137
3095 VS=0:N0=N-1:V0=0:SH=-1:RETURN
                                    :rem  63
3100 GOSUB750                       :rem 224
3110 INP$=RIGHT$("00"+MID$(STR$(F(N,0)),2
     ),2):XX=XV(0):YY=N+2-Y0:GOSUB300
                                    :rem 182
3120 FORV=1TO3                      :rem  71
3125 GOSUB3500                      :rem  19
3130 INP$=F$:XX=XV(V):GOSUB300:NEXT:RETUR
```

Music Editor per Commodore 64

```
     N                              :rem 213
3200 GOSUB650:INP$=MID$(SP$(N,V),7):GOSUB
     300:RETURN                     :rem  99
3500 F$="{ 3 SPAZI}":IFF(N,V)=0THENRETURN
                                    :rem  95
3505 IFF(N,V)=-1THENF$="***":RETURN
                                    :rem  16
3510 I=57+INT(LOG(F(N,V)/F0)/LOG(2↑(1/12)
     )+.5)                          :rem 245
3520 J=(I-(INT(I/12)*12))*2+1:F$=MID$(N$,
     J,2)+MID$(STR$(INT(I/12)),2):RETURN
                                    :rem 178
3600 INP$="":INPUT#2,INP$:GOSUB3900:RETUR
     N                              :rem  75
3900 SP$(N,V)=LEFT$(SP$(N,V),6):SP$(N,V)=
     SP$(N,V)+INP$:RETURN           :rem 114
4000 GOSUB2500:IFBR$=""THENRETURN
                                    :rem  28
4010 PR=1:OPEN1,4:CMD1:RESTORE:READWR$
                                    :rem 146
4020 PRINTCHR$(10)SPC(20)CHR$(18)WR$CHR$(
     146)": "CHR$(14)BR$CHR$(15):rem 185
4030 PRINTCHR$(10)CHR$(10):GOSUB4500
                                    :rem  61
4040 PRINTCHR$(18)" N.{ 2 SPAZI}DUR.
     { 2 SPAZI}V.1{ 2 SPAZI}V.2
     { 2 SPAZI}V.3"CHR$(146)        :rem 248
4050 PRINTCHR$(10)                  :rem   9
4060 FORN=1TOSZ:IFF(N,0)=0THEN4100
                                    :rem  40
4070 PRINTRIGHT$("000"+MID$(STR$(N),2),3)
     ;                              :rem  77
4080 PRINTSPC(2)RIGHT$("00"+MID$(STR$(F(N
     ,0)),2),2)"{ 4 SPAZI}";        :rem 189
4090 FORV=1TO3:GOSUB3500:PRINTF$"
     { 2 SPAZI}";:NEXT:PRINT:NEXT
                                    :rem  57
4100 PR=0:PRINT#1:CLOSE1:RETURN :rem 234
4500 PRINTSPC(48)CHR$(18)"V.1{ 2 SPAZI}V.
     2{ 2 SPAZI}V.3"CHR$(146)       :rem   7
4510 FORN=1TO5:READWR$:PRINTSPC(40)CHR$(1
     4)WR$CHR$(15)": ";:FORV=1TO3
                                    :rem 122
4520 GOSUB2600:PRINTSP$"{ 2 SPAZI}";:NEXT
     :PRINTCHR$(10);:NEXT           :rem 199
4530 V=1:FORN=6TO9:READWR$:PRINTSPC(40)CH
     R$(14)WR$CHR$(15)": ";:GOSUB2600
                                    :rem  74
4540 PRINTSP$"{ 2 SPAZI}":NEXT:PRINTCHR$(
     10)CHR$(10):RETURN             :rem  37
8000 RESTORE:PRINT"{CLR}";:FORI=1TO22:PRI
     NT"B"TAB(23)"B"TAB(38)"B":NEXT
                                    :rem  22
8010 PRINT"{HOME}{GIU'}[<Q>]{ 22 *}[<W>]
     { 10 GIU'}"                    :rem  70
8020 PRINT TAB(23)"[<Q>]{ 14 *}[<W>]
     { 9 GIU'}"                     :rem 159
8030 PRINT"[<Z>]{ 22 *}[<E>]{ 14 *}[<X>]"
                                    :rem 183
8040 PRINT"{HOME}"TAB(6)"{RVS}DUR."TAB(11
```

```
      )"V.1"TAB(15)"V.2"TAB(19)"V.3";:READ
      WR$                           :rem 115
8050  PRINT TAB(24)"{RVS}"WR$":{ 2 GIU'}":
      FORI=1TO9:READWR$:PRINT"{RVS}"TAB(24
      )WR$:NEXT                     :rem 115
8060  PRINT"{GIU'}";:FORI=1TO9:READWR$,WTS
      :PRINT"{RVS}"TAB(24)WR$": "WTS:NEXT
                                    :rem 251
8070  RETURN                        :rem 175
9000  DATA"BRANO","DC1","DC2","W
      { 2 SPAZI}","A/D","S/R","C1 ","C2 ",
      "FON","LEV"                   :rem 140
9010  DATA"F1 ","EXECUTE","F2 ","SAVE","F3
       ","LOAD","F4 ","PRINT"       :rem   9
9020  DATA"F5 ","SLOW","F6 ","FAST","F7 ",
      "STOP","F8 ","SHIFT","CLR","CLEAR"
                                    :rem 138
9040  DATA 6,2,11,3,15,3,19,3,"C C#D D#E F
       F#G G#A A#B "                :rem 158
9050  DATA"270110","270303","310303","3503
      03","270403","310403"         :rem 105
9060  DATA"350403","270503","310503","3505
      03","270603","310603","350603"
                                    :rem  25
9070  DATA"270703","310703","350703","2708
      01","270903","271003","271103"
                                    :rem  35
10000 REM SYS 49152 = SCROLL UP     :rem 138
10010 REM SYS 49227 = SCROLL DOWN
                                    :rem  33
10020 DATA169,080,133,251,169,120,133,253
      ,169,004                      :rem 152
10030 DATA133,252,133,254,162,000,160,000
      ,177,253                      :rem 133
10040 DATA145,251,200,192,023,208,247,024
      ,165,251                      :rem 145
10050 DATA105,040,133,251,165,252,105,000
      ,133,252                      :rem 128
10060 DATA165,253,105,040,133,253,165,254
      ,105,000                      :rem 139
10070 DATA133,254,232,224,019,208,215,160
      ,000,169                      :rem 146
10080 DATA032,145,251,200,192,023,208,249
      ,234,076                      :rem 152
10090 DATA146,192,234,234,234,169,072,133
      ,253,169                      :rem 172
10100 DATA032,133,251,169,007,133,252,133
      ,254,162                      :rem 142
10110 DATA000,160,000,177,251,145,253,200
      ,192,023                      :rem 126
10120 DATA208,247,056,165,251,233,040,133
      ,251,165                      :rem 151
10130 DATA252,233,000,133,252,056,165,253
      ,233,040                      :rem 137
10140 DATA133,253,165,254,233,000,133,254
      ,232,224                      :rem 141
10150 DATA019,208,214,160,000,169,032,145
      ,253,200                      :rem 138
10160 DATA192,023,208,249,234,234,169,080
      ,133,251                      :rem 161
10170 DATA169,216,133,252,162,000,160,000
      ,165,255                      :rem 144
10180 DATA145,251,200,192,023,208,247,024
      ,165,251                      :rem 150
10190 DATA105,040,133,251,165,252,105,000
      ,133,252                      :rem 133
10200 DATA165,253,105,040,133,253,165,254
      ,105,000                      :rem 135
10210 DATA133,254,232,224,019,208,215,160
      ,000,165                      :rem 138
10220 DATA255,145,251,200,192,023,208,249
      ,096,000,9999                 :rem 164
```

R·U·N

Recensioni Software

a cura della redazione

TRENINO cod. CCDMV01

Dal film "Avalance Express" direttamente sul VIC 20

Non ditemi che non avete mai visto quel bis-bis-nonno dei nostri attuali; giochi, costituito da tante tesserine, numerate dall'1 al 15, che andavano ordinate rispettando il principio della consequenzialità. No? Ok, siete troppo giovani!
Si immagini ora di trasporre il tutto in un'immagine video, di allargare considerevolmente la "scacchiera", di costruire casualmente sopra ogni tesserina un tratto di strada ferrata (curve a sinistra, curve a destra, rettilinei, incroci...) e di piazzarci su un treno senza conducente. Direte "Beh, non è rimasto molto del gioco precedente!".
In effetti, è proprio così , anche perché i movimenti per il controllo sono completamente diversi: mentre nel Gioco del 15, per esempio, per completare la configurazione di figura 1, sarebbe necessario spostare fisicamente le tesserine, qui su video quello che noi possiamo spostare è la "casella bianca".
Si nota che spostando la casella bianca, la tesserina immediatamente alla sinistra scambia la sua posizione con la casella bianca.
Analogo discorso vale per tutte e quattro le direzioni ammesse (anche nel Gioco del 15 è impossibile andare in diagonale): se si sposta la casella bianca in giù di una posizione, quello che c'era sotto di lei ora passa sopra. Semplice, vero? Come sempre però tra il dire e il fare c'è di mezzo il ...mare, anzi qui c'è di mezzo un treno impazzito che avanza inesorabilmente sul percorso.
Scopo del gioco sarà dunque quello



Ecco come si presentano due vidiate di questo game



di "costruirgli" davanti la strada ferrata prima che quest'ultimo possa deragliare.
Poiché esiste la possibilità di formare un percorso chiuso e quindi di far "ciclare" indefinitamente il treno, si è reso necessaria l'introduzione di un tempo limite: dopo 10 minuti, qualunque sia il CREDIT, il programma interrompe e conclude automaticamente il gioco.

Esistono tre livelli di difficoltà (basso, medio, alto) in corrispondenza dei quali si hanno tre differenti velocità del treno (se però quella più alta non vi sembra sufficiente, modificate nella linea 18 A=3 con A=01. Questa solo per dare un'idea delle potenzialità del gioco!).
Il gioco gira sul VIC 20 nella sua configurazione minima di memoria (3,5 Kbyte utili) e non ha dunque bisogno di alcuna espansione; il controllo dei movimenti può essere eseguito indifferentemente con la tastiera o con il joystick.
Il programma si compone di una parte BASIC (circa 800 byte, per le istruzioni e la creazione casuale del tempo di gioco) e di oltre 2600 byte di linguaggio macchina (non visibile, ovviamente).
La velocità, una grafica chiara ed essenziale (si noti come a tratti di pista uguali corrispondano colori uguali), il suono, l'impegno richiesto, fanno di questo gioco, tutt'altro che banale, un efficace esempio di quanto si possa realizzare in soli 3,5 Kbyte di memoria.

SIRPENT cod. CDBMV01

Un sinuoso arcade game per VIC 20

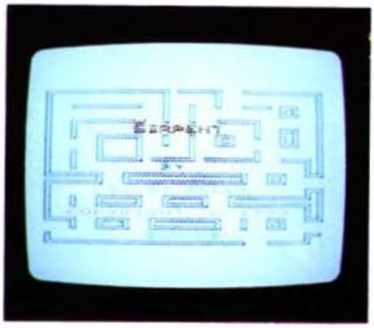
Eccoci alle prese con un vorace serpentello di alto lignaggio (si noti la corona nobiliare) che, risucchiato da un notissimo video-gioco da bar, attraverso opportuni adattamenti, è chiamato a far da protagonista in un nuovissimo ed avvincente gioco per uno tra i più diffusi personal computer del mondo: il VIC 20.
Ingordo come pochi altri, il nostro serpentello, si aggira affamato per i corridoi del labirinto nel tentativo di far piazza pulita delle prede ivi disseminate. Ma, ahimè, è anche un serpentello dalla vista un po' corta. Se lasciato libero, dopo un po' continua a ciclare indefinitamente, senza accorgersi di es-

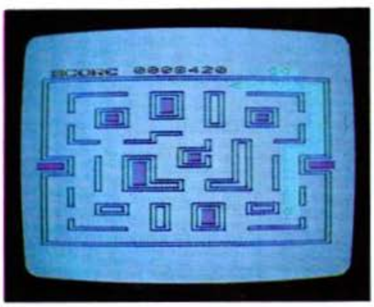
R·U·N

Recensioni Software

sere già passato varie volte per la stessa strada. Sei in grado di dargli una mano, ovvero riesci a forzarlo a mangiucchiare tutte le prede (i quadratini concentrici), così che, pulito il campo, possa passare allo schermo successivo?
Stai attento però: ogni preda frutterà sì 1000 punti, ma farà aumentare di due caselle il corpo del serpente. Bisognerà dunque "guidarlo" in modo tale che non riesca a morsicarsi la coda o il corpo: i denti avvelenati, infatti, ai lati della lingua biforcuta, lo porterebbero istantaneamente alla morte, né sarà possibile fargli invertire la direzione di marcia (il classico dietro-front) perché i corridoi in cui si è andato a cacciare sono tremendamente stretti.
Sembrerebbe davvero semplice, ma, vi assicuro, la difficoltà esiste: il serpente infatti, appena lasciato "libero", riprende di nuovo a far di testa sua e, fondamentale, una volta arrivato ad un bivio, sceglie subito una sua direzione preferenziale.
Se a questo punto decidete di farlo girare dalla parte opposta (si badi bene: è questione di millisecondi), avrete il classico "crash" dovuto alla vietata inversione di marcia.







È necessario dunque premere il tasto corrispondente alla direzione voluta prima che il serpente arrivi al bivio, proprio perché, una volta al bivio, il nostro testardo serpentello ha già deciso dove andare e non vuol sentire ragioni di sorta. Occhio pronto, dunque, per anticiparlo sempre ad ogni istante prima di un bivio.
"Pappate" tutte le prede, si passa allo schermo successivo ma, (e qui viene il bello) questo ne contiene una di più del precedente; per terminare dunque questo schermo bisognerà mangiare una preda in più, con un ulteriore allungamento del serpente di due caselle.
Si parte al primo schermo con sette prede da "mangiucchiare", passando poi alle otto del secondo schermo e così via. D'accordo, (parlo coi VIC-dipendenti) all'inizio vi sembrerà una passeggiata ma, vi assicuro che arrivare a superare i 400.000 punti richiederà una buona dose di destrezza e di allenamento, soprattutto per sfruttare convenientemente le "naturali" direzioni di marcia. Il serpente, infatti, arriverà ad avere un corpo di oltre 60 caselle e sarà impossibile tenerlo tutto sul corridoio più esterno del labirinto. Ciò vi obbligherà ad elaborare strategie in tempo reale (come per esempio fargli mangiare prima i quadratini che si trovano al centro dello schermo, piuttosto che quelli sui corridoi periferici), nel tentativo di battere il record ufficiale che, per il momento, ammonta a 465.550 punti (circa 15 minuti di gioco e varie dita piagate).
Il gioco gira sul VIC 20 nella sua configurazione base di memoria (3583 byte utili) ed è stato ovviamente realizzato in linguaggio macchina; si compone di due programmi distinti: il primo (SIRPENT 1), comprende la presentazione, le istruzioni di gioco (in italiano visto che è made in Italy) e i comandi da usare (è possibile giocare indifferentemente con la tastiera o col joystick). Il secondo (SIRPENT 2) viene automaticamente caricato dal primo e contiene il gioco vero e proprio (una volta note le regole del gioco è comunque possibile caricare solo il secondo programma).
Grafica, colore, velocità, suono, non sono che poche qualità di questo sorprendente gioco, capace di tenervi incollati per ore e ore allo schermo, magari in un'incredibile sfida tra gli amici.

I programmi in vendita presso il TechnoClub a L. 30.000 cad. (iva inclusa) possono essere ordinati utilizzando il coupon pubblicato a pag. 66.

PRINT

Recensioni Libri

COMMODORE 64 Il BASIC

di Rita Bonelli



È il primo di una serie di libri di prossima pubblicazione scritti sempre dalla stessa autrice e dedicati al popolare C 64.
Questo primo volume, di taglio introduttivo, sarà sicuramente apprezzato dalle numerose persone che si accostano per la prima volta al mondo dell'informatica individuale.
Anche i più esperti potranno utilizzarlo come efficace strumento di consultazione.
Una prima parte del libro è una introduzione generale alla programmazione. Vengono poi esposti i concetti principali del BASIC e quindi, mediante l'ausilio di parecchi programmi dimostrativi, vengono illustrate le varie istruzioni del linguaggio.
Seguono alcuni capitoli che spiegano il funzionamento della tastiera, delle stampanti e delle unità a nastro e disco per la memorizzazione di programmi e dati.
L'ultima parte del volume fornisce alcune utili indicazioni sulla costruzione di programmi e sull'utilizzo della memoria.
Le due appendici conclusive sono dedicate all'esame della tastiera e alla presentazione del PETSPEED 64, un compilatore BASIC che permette di migliorare notevolmente la velocità di esecuzione dei programmi.

Voi e il vostro Commodore 64

di Fulvio Francesconi e Fernando Paterlini



Finalmente un testo che affronta in modo accattivante e quasi divertente lo studio di un personal computer.
Il libro affronta in modo graduale la programmazione del Commodore 64 e il carattere spesso scherzoso non riduce la rigorosità dell'esposizione.
Si tratta di un testo chiaro, facile da leggere e adatto veramente a tutti.
Ad una parte introduttiva teorica, segue una serie di capitoli che illustrano in dettaglio come programmare e utilizzare il calcolatore.
Per ogni argomento affrontato, vengono presentati dei programmi che costituiscono una utilissima forma di esercitazione. Alcuni capitoli specifici illustrano le caratteristiche più interessanti del calcolatore, quali la grafica e il suono.
Il libro si conclude con un esame delle periferiche più utilizzate (stampante, disco, cassette), e dei programmi più popolari.
Nel volume non manca un originale glossario che abbina definizioni tecniche ad altre più spiritose e divertenti.

Alla scoperta del VIC 20

di Rita Bonelli e Daria Gianni



Questo volume prosegue e completa il discorso sul VIC 20, iniziato dalla stessa autrice Rita Bonelli con il libro "Vogliamo Incominciare Così".
Mentre il primo era un testo introduttivo appositamente concepito per i principianti, questo è stato studiato soprattutto per quanti intendono approfondire maggiormente le proprie conoscenze della macchina.
Suddiviso in 8 capitoli e fornito di 9 appendici, il libro si sofferma principalmente sulla architettura del sistema e del microprocessore 6502.
Un altro capitolo è dedicato alla gestione del video e in particolare all'utilizzo della grafica.
Uno degli argomenti che è stato maggiormente approfondito è quello relativo ai dispositivi di I/O. Oltre ad uno studio dell'hardware di ingresso e uscita del VIC, vengono presentati in dettaglio i principali dispositivi quali stampante, cassetta e disco.
Completa il volume, una serie di appendici che presentano le varie cartridge disponibili.
Si tratta in definitiva di un testo molto curato e di agevole lettura e consultazione.
I vari capitoli sono corredati di numerose figure e parecchi programmi che agevolano la comprensione degli argomenti trattati.

Per l'acquisto vedi coupon a pag. 66.

Personal e home computer

Il manuale base per l'uso del VIC 20

Rita Bonelli
Daria Gianni

Alla scoperta del VIC 20 architettura e tecniche di programmazione

Un libro atteso da quanti - e sono moltissimi-hanno acquistato uno dei Personal Computer del giorno: il VIC 20 Commodore.

Naturale completamento del precedente "Impariamo a programmare in BASIC con il VIC/CBM", questo manuale può soddisfare diverse esigenze.

Ci sono capitoli che trattano i file su disco e cassetta, la stampante VIC 1515, alcuni cartridge come VIC STAT, VIC GRAF, SUPER EXPANDER. Un'intera parte è dedicata alle porte I/O, al chip d'interfaccia video, al linguaggio macchina del calcolatore. Un'ultima importante annotazione: tutti i programmi che compaiono nel testo sono stati provati sul calcolatore e sono disponibili su cassetta e floppy disk.

300 pagine
Lire 22.000
Codice **338 D**



GRUPPO EDITORIALE JACKSON

Attenzione compilare per intero la cedola
ritagliare (o fotocopiare) e spedire in busta chiusa a:
GRUPPO EDITORIALE JACKSON
Divisione Libri
Via Rosellini, 12 - 20124 Milano

CEDOLA DI COMMISSIONE LIBRARIA

VOGLIATE SPEDIRMI

n° copie	codice	Prezzo unitario	Prezzo totale
	338D	L. 22.000	

Desidero anche i programmi su:
☐ Floppy disk a L. 25.000
☐ cassette a L.15.000

☐ Pagherò contrassegno al postino il prezzo indicato più L. 2000 per contributo fisso spese di spedizione.

Condizioni di pagamento con esenzione del contributo spese di spedizione

☐ Allego assegno della Banca ____________

n° ____________

☐ Allego fotocopia del versamento su c/c n. 11666203 a voi intestato

☐ Allego fotocopia di versamento su vaglia postale a voi intestato

Nome

Cognome

Via

Cap   Città   Prov.

Data   Firma

Spazio riservato alle Aziende. Si richiede l'emissione di fattura

Partita I.V.A. |_|_|_|_|_|_|_|_|_|_|_|

nuovidea

Scatena la poten
con il nuovo plotter

Scrive, disegna, fa i grafici, stampa a



Caratteri
di diverso formato.



Figure geometriche.



Nuovo! Dalla Commodore. È arrivato il primo plotter/stampante che puoi collegare direttamente al tuo VIC 20, o al Commodore 64.

Provalo, e scatena tutta la potenza del tuo computer... guarda quante cose sa fare.

Disegna a 4 colori, figure anche complesse; e sa tracciare disegni molto grandi, di 30-40 cm. Fa i grafici.

Ha una grafica da plotter estremamente accurata con una altissima risoluzione (di 0,2 mm) perché usa 4 pennini che stampano formando una linea continua.

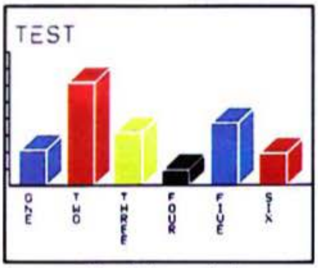
È una vera stampante! Stampa i normali caratteri in quattro differenti misure automaticamente.

Con opportuni comandi sul computer, stabilisce le dimensioni dei caratteri, le maiuscole o le minuscole.

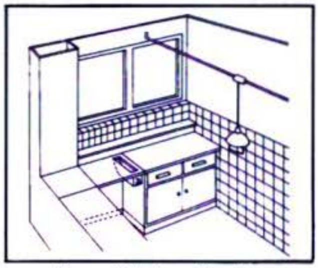
Ma non solo. Può stampare sia in orizzontale - fino a un massimo di 80 caratteri per riga - che in verticale usando

za del tuo VIC 20 /stampante a colori.

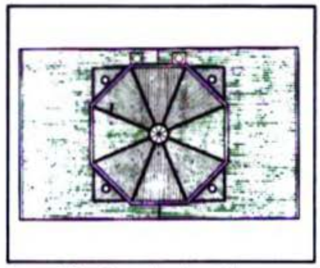
4 colori (anche col Commodore 64).



Grafici a colori, istogrammi.



Progetti d'ambiente.



Piante, planimetrie, schemi tecnici.



l'altezza della carta!
Così potrai scrivere testi di qualsiasi "giustezza" tipografica. È facile da usare: è sufficiente un po' di familiarità con il linguaggio BASIC e con i principi della programmazione; e potrai scrivere tu stesso i programmi per il tuo plotter.

E poi confronta il prezzo: il concorrente più vicino costa almeno tre volte tanto... e il nuovo plotter Commodore è anche più compatto e più leggero.
Non c'è miglior regalo che puoi fare al tuo sistema.
Commodore Italiana S.p.A.
Tel. (02) 618321.

commodore
COMPUTER



Libri firmati JACKSON



La Biblioteca che fa testo

DAL TRANSISTOR AL MICROPROCESSORE
La moderna circuiteria a stato solido, la sua evoluzione, le sue prospettive.
80 pag. L. 7.500
Cod. 141A

LA SOPPRESSIONE DEI TRANSITORI DI TENSIONE
Cause, effetti, rimedi ai danneggiamenti, dei transitori d'alta tensione.
224 pag. L. 12.000
Cod. 611H

MANUALE DI OPTOELETTRONICA
Valido supporto per i progettisti elettronici. Questo manuale si compone di numerosi fogli-dati, guide di scelta e tabelle comparative.
208 pag. L. 15.000
Cod. 613P

CAPIRE I MICROPROCESSORI
Spiegazione in forma chiara e dettagliata di come funzionano i microprocessori, le memorie ROM, RAM e le interfacce.
126 pag. L. 10.000
Cod. 342A

MICROPROCESSORI AL SERVIZIO DEL MANAGEMENT
CAD/CAM e robotica la loro applicazione in Azienda, l'impatto su qualità e produttività, le prospettive.
292 pag. L. 20.000
Cod. 335H

LE COMUNICAZIONI RADIO IN MARE
Come orientarsi grazie alla moderna strumentazione e ai suoi codici.
200 pag. L. 15.000
Cod. 706A

ELEMENTI DI TRASMISSIONE DATI
Un valido ausilio per tecnici e studenti che vogliono approfondire le tecniche di comunicazione.
178 pag. L. 10.500
Cod. 316D

GRUPPO EDITORIALE JACKSON

Attenzione compilare per intero la cedola
ritagliare (o fotocopiare) e spedire in busta chiusa a:
GRUPPO EDITORIALE JACKSON
Divisione Libri
Via Rosellini, 12 - 20124 Milano

CEDOLA DI COMMISSIONE LIBRARIA

VOGLIATE SPEDIRMI

n° copie	codice	Prezzo unitario	Prezzo totale
		Totale	

☐ Pagherò contrassegno al postino il prezzo indicato più L. 2000 per contributo fisso spese di spedizione

Condizioni di pagamento con esenzione del contributo spese di spedizione:

☐ Allego assegno della Banca ____________
n° ____________

☐ Allego fotocopia del versamento su c/c n. 11666203 a voi intestato

☐ Allego fotocopia di versamento su vaglia postale a voi intestato

Nome
Cognome
Via
Cap Città Prov.
Data Firma

Spazio riservato alle Aziende. Si richiede l'emissione di fattura

Partita I.V.A. | | | | | | | | | | | |

ORDINE MINIMO L. 50.000

input·output

Piccoli annunci

Cambio, vendo programmi per Commodore 64, easy script, BASIC 4.0, screen graphic, simon's BASIC, monitor ass., 80 colonne, disk blackup, disk manager totosistemi videogame. Scrivere o telefonare a:
Saverio Pensabene - Lungomare C. Colombo, 3544 - 90149 Palermo (Addaura) - Tel. 091/451425

Vendo software per CBM 64 tutti in linguaggio macchina e con ottima grafica per informazioni o richiedere la lista telefonare o scrivere a:
Alessandro Palma - Via A. Longo, 50 - 80127 Napoli - Tel. *81/656305

Compro per CBM 64 programmi di idraulica e termotecnica. Cerco inoltre programma traduttore di CW e RTTY (solo ricezione). Rispondo a tutti.
Dino Fornaciari - Villaggio Dante, 30 - 52100 Arezzo - Tel. 0575/351415 (ore pasti)

Cambio programmi su cassetta per Commodore VIC 20.
De Faveri Anna - Via Longhena, 58— 30175 Venezia - Tel. 041/936146

Cambio, vendo per CBM 64 vasta gamma giochi ed utility. Richiedere ed inviare liste. Disponibili sia su cassetta che su disco. Telefonare serali a:
Paolo Raimondo - C.so Vittorio Emanuele, 71 - 10128 Torino - Tel. 011/545625

Cambio cartuccia gioco "mission impssible" (per il computer VIC 20) da giocare con la tastiera, con qualsiasi cartuccia che si possa giocare senza il registratore e con i tasti F1-F2 ecc. Telefonare il sabato o domenica a:
Giovanni Schillaci - Via Rimembranza, 51 - 93010 Milena (CL) - Tel. 0934/933015

Cambio, vendo programmi per il VIC 20 possibilmente in linguaggio macchina. Cerco programmi in LM per il CBM 64. Inviatemi la vostra lista. Vendo Simon's BASIC 64 disco o cassetta a L. 40.000 con il relativo manuale.
Roberto Oselladore - Via Passo S. Boldo, 35/2 - 30030 Favaro Veneto (VE) - Tel. 041/631106

Vendo corso BASIC prima parte interamente in italiano. Originale Commodore per VIC 64. Comprendente due cassette più un manuale; tratto solo con BARI.
Nicola Lavopa - Via T. Cardarelli, 22 - 70125 Bari - Tel. 472842

Compro, cambio, vendo programmi per C64 prezzi modici gestionali L. 40.000 utilizzo L. 25.000 giochi L. 10/15.000.
Fabio La Rotonda - Via E. Capocci, 27- Napoli - tel. 081/202855

Vendo su cassetta i seguenti giochi per VIC 20: alien-blitz, abdactor, amok, grid. runner, vic-panic, sci, medioevo, paroliamo e terror; (quasi tutti per joystick) a L. 30.000. Telefonare ore pasti, massima serietà.
Alberto Artioli - Via M. Marzabotto, 4 - 45030 S. Maddalena (RO) - Tel. 0425/757672

Vendo per VIC 20 e CBM 64 rispettivamente 20 ed 8 giochi come al bar a L. 30.000, per 64 Simon's BASIC, BASIC 4.0, Pet speed, 80 colonne con istruzioni L. 30.000, gestionali fatturazione magazzino, data base word processing, the last one.
Claudio Giovanelli - Via Ripamonti 194 - 20141 Milano - tel. 02/536923

Compro, cambio, vendo programmi per CBM 64, di vario genere. Dispongo di splendidi giochi e di ottimi programmi di utilità richiedere la lista inviando la vostra.
Vincenzo Sena - Via Ferrovia, 26 - 80033 Cicciano (NA)

Per VIC 20 ho una nostroteca di circa 100 programmi (videogiochi, didattici, gestionali, giochi di società, grafica, per la casa, ecc.) che vendo a prezzo da sballo. Ricco e dettagliato elenco inviando L. 1.000 anche in francobolli.
Giovanni Vermiglio - Viale Friuli, 27 - 10015 Ivrea (TO)

Vendo per CBM 64 Simon's BASIC su nastro completo di manuale in italiano e programmi dimostrativi occasione unica in italiano e programmi dimostrtivi occasione unica aggiunge 114 comandi al tuo computer telefonare ore serali a:
Carlo Ferrari - Via G. Martinelli - 35044 Montagnana (PD) - Te. 0429/82469

Cambio software per VIC 64 preferibilmente su disco. Inviare listati o telefonare a:
Fernando Forner - Via Valperga Caluso, 21 - 10125 Torino - Tel. 011/6506538

Vendo ottimo software per VIC 20: videogame e utilitie 48 programmi su cassetta; inoltre 52 per ZX81 e 100 per ZX Spectrum in L.M.
Armando Mazza - Via Settembrini, 96 - 70053 Canosa (BA) - Tel. 0883/64050

Vendo programmi per VIC 20 a prezzi eccezionali; sei un principiante? Ti invierò una cassetta in contrassegno a L. 15.000 contenente 35 programmi di giochi utili e dimostrativi.
Gianluca Burattelli - Via G. Di Vittorio, 2 - 58022 Follonica (GR) - Tel. 40903

Cambio numerosi programmi in L.M. per VIC 20 (espanso e non), tromb of drewan, arcadia, wacky vaiters, panic, subchase, metagalatic llamas, gidrunner, boss, bonzo, pac-man, skramble, kosmic kami kaze, e molti altri. Possiedo inoltre metoo per la copia di alcune (sargon II, jelly monsters, programmer's aid, etc.) mi interessano coiatori per nastri e ogni genere di programmi in L.M. gradisco anche adventures di goni tipo. Mandatemi le vostre liste, vi manderò le mie. Rispondio a tutti.
Luca Alebardi - Via Longari, 21 - 24010 Ponteranica (BG) - tel. 035/572183

Vendo Commodore CBM 4016, 16 kbyte, monitor 11" pollici fosfori verdi incluso + registratore C2N + manuale BASIC 4.0 + manuale Jackson. Il tutto a L. 650.000. Telefonare dopo ore 20 a:
Sergio Giannessi - Via Ovada, 19 - 20142 Milano - Tel. 02/810868

Vendo VIC 20 completo di alim. modul. TV interf. reg. VIC-1211 asup. exp. stamp. VC-1515, cartuccia Sargonchess, manuali orig. ingl. e ital., guida EVM - VIC 20, cassetta giochi, graf. di funzioni con stampa, dis. h. ris. miglior offerente in blocco o separatamente.
Paolo Berbellini - Via Roma, 18 - 63023 fermo (AP) - Tel. 0734/32136

Vendo giochi su nastro a metà prezzo per Commodore 64. Per informazione telefonare dopo le ore 20 a:
Ferruccio Zanardi - Via Siracusa, 3 - 37138 Verona - Tel. 565609

Vendo calc-result advanced (in italiano) originale. Visicalc like per CBM-64 L. 200.000 (prezzo originale L. 400.000). Regalo inoltre all'acquirente Simon's BASIC.
Alessandro Fogar - Via Venezia, 26 - 34073 Grado - Tel. 0481/768655

Vendo VIC 20 nuovo poco usato causa disinteressamento materia con registratore disponibile a fine gennesio + cassetta gorf + manuale d'istruzione a L. 400.000 + trasformatore.
Guecello Di Porcia - Via Actinate, 141 - 35100 Padova -Tel. 049/45389

Vendo per CBM-64 "frogger-64" (su cassetta) "grammaster" (scacchi 9 livelli di difficoltà - su disco) "screen-graphics-64" (24 comandi in più per gestire grafica ad alta risoluzione con demo e istruzioni - su disco).
Stefano Vandelli - Via Parini, 5 - 41013 Castelfranco E. (MO) - Tel. 099/924259

Cambio programmi software per Commodore 64. Mandare lista di programmi a:
Franco Bruno - Via Giorgio Bratti, 100 - 47023 Cesena (FO) - Tel. 0547/23810

Cambio per Commodore 64 programmi di tutti i tipi. Ma specialmente giochi.
Arturo Parodi - Via S. Defendente, 44/b - 15011 Acquiterme (AL) - Tel. 57280

Cambio sistema Commodore 64 + data Sette C2N + plotter/stampante 4 colori + molti programmi di grafica e giochi tutto nuovissimo. Cambio con Apple + IIe + disck drive o con Apple compatibile + disck drive + monitor.
Davide

Vendo o cambio molto software per VIC 20 (giochi/utility) sia in BASIC che linguaggio macchina solo su cassetta.
Fabio Siani - Via Buonarroti, 19 - 20149 Milano - tel. 02/4694089

Per la pubblicazione dei vostri annunci utilizzare il coupon a pag. 66

input·output

Piccoli annunci

Vendo VIC 20 + registratore C2N + espansione 8 kbyte + software 40 colonne + altri programmi e cavetti L. 250.000 + postali.
Alberto Del Bianco - Via Trasimeno, 7 - 52100 Arezzo - Tel. 300692

Vendo programmi per VIC-20 LM: frog gridrunner - abductor - alien blitz - raid on isran troit altri (13k). Vendo 16K Ram + 8KRam + toolkit + hi - res + programmatore di esprom + L e Cartridge su nastro
Nicola Pedroli - Via Vigevano, 36 - 20010 Bareggio (MI) - Tel.02/9013314

Vendo, cambio giochi per C64 oppure scambio programmi + vendo primi 10 numeri di video giochi + 1 cartuccia "visibol solar system".
Angelo Settembrini - Via Cassanese, 194 - 20090 Segrate (MI) - Tel. 02/2136514

Vendo, scambio giochi e software per CBM 64 e VIC 20 (memoria base). Per informazioni telefonare.
Paolo Tognana - Via Guidi, 1 - 35100 Padova - Tel. 049/683552-687140

Vendo o cambio programmi su cassetta o listati per VIC 20.
Telefonare ore serali a:.
Paolo Malpeli - Via Coppellotti, 20 - 29100 Piacenza - Tel. 0523/71341

Vendo per VIC 20 cassetta C 90 contenente 98 giochi: giochi utility ecc., molti in LM; vendo, scambio e compro giochi per C 64
Giulio Ravagni - C.so Rosmini, 63* - 38068 Rovereto - Tel. 0464/34475

Vendo programmi per VIC 20. Inviatemi 1.000 lire in francobollli è vi spedirò la mia lista di programmi (circa 500), scambio anche su disco o cassetta.
Enrico Sturaro - C.so Casale 416/6 - 10132 Torino - Tel. 011/898756

Vendo e cambio per Commodore 64 circa 400 programmi tra giochi e utility. Tutte le ultime novità da tutto il mondo! Richedere lista a:
Claudio Barchesi - Via S. Martino, 10 - 60035 Jesi (AN) - Tel. 0731/3347

Compro, cambio e vendo programmi per CBM 64. Forza commodoriani annuncio sempre valido. Scrivete scrivete scrivete......... a:
Computer Club CBM 64 - 20062 Cassano D'Adda (MI)

Commodoriani è nato il Computer Club CBM 64. Perscambio acquisti ecc......
Programmi vari inviare richieste offerte - annuncio srmpre valido scrivere alla casella postale n. 4 - 20062 Cassano D'Adda (MI)

Compro, vendo, scambio programmi per VIC 20 invio liste a tutti, invio inoltre cassetta con programmi achi mi invia una cassetta con I suoi assicuro risposta a tutti in qualsiasi caso.
Roberto Salerno - Via G. Galilei, 15 - 87036 Quattromiglia Rende (CS) - Tel. 839360

Per Commodore 64 scambio programmi, giochi e utilities. Cerco inoltre software di termotecnica.
Dino Fornaciari - Villaggio Dante, 30 - 52100 Arezzo - Tel. 0575/351451

Vendo estensione Basic per Commodore 64 e gioco scacchi a sole L. 50.000. Scambio anche oltre 80 programmi di gestione file/utility/giochi. Il tutto in disco o cassetta.
Alessandro Bruscagin - S. Angelo, 188 - 31100 Treviso (TV) - Tel. 56773

Commodore 64 cerco programmi da acquistare su cassetta a prezzi modici, inviare lista e modalità di pagamento.
Stefano Valli - Via Luigi Cibrario, 17 - 47031 Repubblica di San Marino

Scambio o vendo a prezzo minimo programmi gestionali, utility e giochi su disco o cassetta per Commodore 64, invio gratuitamente elenco. I programmi sono completi di istruzioni.
Franco Viasetti - Via S. Frigerio, 11 - 25100 Brescia - Tel. (030) 53162

Scambio o vendo per Commodore 64 più di 300 programmi su disco o cassetta - massima serietà. Scrivere a:
Luigi Beviglia - Casella posta 41 - 21052 Busto Arsizio

Si è costituito a Bari il Pocket Group Club utenti Commodore Puglia. Per i soci sono previste numerose facilitazioni: dell'acquisto di materiale commodore a prezzi speciali, all'accesso alla biblioteca software del club composta da centinaia di programmi di ogni tipo. Il Club dispone, inoltre, di servizio di fotocopiatura della vasta biblioteca di manuali programmi, libri e riviste italiane e straniere. Per informazioni scrivere a:
Pocket Group - Via Amoruso, 34 - 70124 Bari

Vendo perCBM 64 vasta gamma di software, giochi utility ecc., la maggior parte in LM a prezzi competitivi. I programmi vengono tutti direttamente dall'America! Garanzia su ogni gico.
Antonio Bartolo - Via Vico Vitetta, 37 - 89100 Reggio Calabria - Tel. 0965/54350 (ore pasti)

Vendo 250 programmi per VIC 20 prezzi bassissimi sia singoli che in blocco. Possiedo giochi come blitz, briscola, bonzo, the castle ecc. Richiedere lista inviando L. 1.000 o vendo le 5 cassette a L. 60.000
Francesco Cadoni - Via G. Matteotti, 38 - 07100 Sassari - Tel. 216489

Compro espansione di memoria 16 kbyte per VIC 20. Telefonare dopo le 20.30 a:
Osvaldo Pennacchi - Via F. Squarcione, 5 - 35100 Padova - Tel. 049/36479

Ho un CBM 64, vorrei acquistare dei programmi gioco con supporto cassetta o listato, per scambi scrivete o telefonare (ore serali) e riceverete informazioni. Per vendita inviate lista, caratteristiche e prezzi.
Paolo Giordano - Viale Brigate Partigiane, 18/S - 16129 Genova - Tel. 010/587762

Vendo VIC 20 + registratore + 16 kbyte + 3kbyte s. exp. + 3kbyte + monitor l. macchina + 2 giochi su cartuccia + trislot + light per + joystick e paddle + cassetta con giochi in l. macchina. Il tutto in buone condizioni + manuale e VIC revealed L. 70.000.
Giuseppe Caggese - Via Francesco Tumiati, 5 - 00128 Roma - Tel. 06/5204306

Compro qualsiasi programma per Commodore VIC 20 e 64, solo se a prezzi modici.Inviare elenchi.
Alessandro Traine - Via Aldo Moro, 15 - 30033 Noale (VE) - tel. 041/441880

Cerco possessori di Commodore 64 per scambio software e informazioni, cerco tra l'altro simon's BASIC da cambiare con altri programmi o da acquistare se a prezzo modico. Inviare liste. Sono in possesso anche di software per il VIC 20.
Paolo Di Mauro - Via Bertieri, 1 - 20146 Milano - Tel. 02/471803

Cambio/vendo software su cassetta per Commodore 64: utility, video game, ecc.
Stefano Lomurno - Via Giambellino, 10 - 20146 Milano - tel. 02/4235248

Cambio, vendo VIC 20 programmi, dispongo di programmi per VIC in espanso, con 3 kbyte di espansione, con 8 kbyte da L. 2.000 a L. 10.000 l'uno tra i quali: rat race, pacman, frogger, crazy cong, scramble... solo Roma
Luca Dalcò - Via N . Tartaglia, 5 - Roma - 805517

Vendo bellissimi programmi di giochi pe VIC 20 inviare 400 lire in francobolli per la lista e i prezzi regalo inoltre ai primi 5 cassetta con vari programmi scrivetemi risponderò a tutti.
Gerardo Ventura - Via Regina Elena, 82 - 65100 Pescara

Cambio, vendo programmi (anche listati) per VIC 20: dispongono di vasto software, giochi (anche il LM), utilitie, routine. Rispondo a tutti. Scrivere o telefonare a:
Maurizio Prandoni - Via N. Sauro, 25 — 20010 Sang Giorgio su Legnano - Tel. 0331/401856

Vendo fotocopie di listati di programmi per il Commodore 64. Per l'elenco inviare L. 500 in francobolli a:
Lauro Michelotti - Via Boboli, 1/A - 51017 Pescia (PT7

Cambio o compro programmi per VIC 20 su nastro o disco.
Sesto Pederzani - Via G. Matteotti, 3/A - 20094 Corsico (MI) - Tel. 02/4408980

Cambio software VIC 20, CBM 64 e Apple IIe. Prego inviare lista programmi. Cedo programmi in cambio di materiale surplus elettronico in buone condizioni.
Giuseppe Marcellino - Viale Cadorna, 8 - 21052 Busto Arsizio - Tel. 632686

Vendo cartucce (cartridge) per il VIC 20 "alien", "avenger", "ratrace" in perfetto stato, usate pochissimo, 1 mese di vita funzionanti a L. 35.000/40.000. Vero affare.
Luca Maccari - Via Pismonte, 5 - 20139 Milano - Tel. 02/5391991

Compro VIC 20 + registratore + videogioco a un prezzo contenuto per ragioni finanziarie. Possibilmente pago a rate. Scrivere o telefonare ore pasti a:
Nicola Branchini - Via Porta Po, 85 - 44100 Ferrara - Tel. 0532/33063

Compro, cambio, vendo per PET CBM 4032 the last one dispongo di numerosi programmi gestionali ingegneria word III giochi originali cursor space invader grafica Command 0 C3 BASIC plus programmatore di EPROM rilocatore codice macchina RTTY + stampa.
Augusto Bernardini - Via Valle Verde, 5 - 05100 Terni - Tel. 56870-47148

GP50A E GP50S

le piccole stampanti per tutti i computer

SEIKOSHA

REBIT COMPUTER
A DIVISION OF GBC



Piccole e compatte dalle prestazioni grandi e generose, le GP50A e GP50S sono realizzate con standard professionali a misura di Personal e Home computer e si impongono quale soluzione ottimale per gli usi hobbystici più disparati a costi incredibilmente sorprendenti. Particolare attenzione merita la GP50S, stampante direttamente interfacciata verso i computer Sinclair ZX81 e Spectrum.

Caratteristiche:

- Stampante ad impatto a matrice di punti da 46 colonne (32 colonne versione GP50S)
- Matrice di stampa 5x8 (7x7 versione GP50S)
- Percorso di stampa monodirezionale (da sinistra a destra)
- Capacità grafiche con indirizzamento del singolo dot
- Possibilità di ripetizione automatica di un carattere grafico
- Velocità 40 caratteri/secondo (35 caratteri/secondo versione GP50S)
- Caratterizzazione: 12 cpi e relativo espanso
- Interfacce: parallela centronics (interfaccia Sinclair versione GP50S)
- Alimentazione carta a frizione (largh. carta fino a 5")
- Stampa 1 originale e 1 copia
- Set di 96 caratteri ASCII
- Consumo 11W (standby) o 17W (stampa)
- Livello di rumore inferiore a 60 dB
- Durata di vita testa: 30 milioni di caratteri
- Peso 1,5 KG
- Dimensioni: 215 (prof.) x 250 (largh.) x 85 (alt.) mm.
- Nastro nero (standard); optionals: rosso, arancio, verde, blu, viola e marrone.

SUPERVIC — LIBRI

CEDOLA DI ORDINAZIONE - LIBRI
da compilare e spedire in busta chiusa a
TechnoClub - Via Rosellini, 12 - 20124 Milano Tel. 6888228

COMMODORE 64 IL BASIC:	cod. AJAC030 L. 23.400
VOI E IL VOSTRO COMMODORE 64:	cod. AJAC023 L. 19.800
ALLA SCOPERTA DEL VIC 20:	cod. AJAC016 L. 19.800

Ordino i seguenti libri per un importo totale di L. .... + L. 2.000
come contributo fisso per spese di spedizione

Cod. .... Cod. .... Cod. ....

Cod. .... Cod. .... Cod. ....

Contanti allegati   Assegno allegato n° ....

Ho spedito l'importo a mezzo vaglia postale

Ho versato l'importo sul ccp. n° 19445204 intestato a Technoclub - Milano

Pagherò in contrassegno al postino al ricevimento dei volumi (valido solo per i soci in Italia)

Nome ....

Cognome ....

Via ....

Città .... C.A.P .... Prov. ....

Se richiesta fattura:
Cod. F. e P. Iva ....

Data ....

Firma ....

Per i soci residenti all'estero — pagamento anticipato (vaglia o versamento su ns. ccp)

SUPERVIC — SOFTWARE

CEDOLA DI ORDINAZIONE - software
da compilare e spedire in busta chiusa a
TechnoClub - Via Rosellini, 12 - 20124 Milano Tel. 6888228

TRENINO:	cod. CCDMV01 a L. 30.000
SIRPENT:	cod. CDBMV01 a L. 30.000

Ordino i seguenti libri per un importo totale di L. .... + L. 2.000
come contributo fisso per spese di spedizione

Cod. .... Cod. .... Cod. ....

Cod. .... Cod. .... Cod. ....

Contanti allegati   Assegno allegato n° ....

Ho spedito l'importo a mezzo vaglia postale

Ho versato l'importo sul ccp. n° 19445204 intestato a Technoclub - Milano

Pagherò in contrassegno al postino al ricevimento dei volumi (valido solo per i soci in Italia)

Nome ....

Cognome ....

Via ....

Città .... C.A.P .... Prov. ....

Se richiesta fattura:
Cod. F. e P. Iva ....

Data ....

Firma ....

Per i soci residenti all'estero — pagamento anticipato (vaglia o versamento su ns. ccp)

SUPERVIC INPUT/OUTPUT

La rubrica INPUT/OUTPUT è gratuita ed aperta a tutti i lettori. Chi desidera comprare, vendere o cambiare hardware o software può inviare il tagliando a J.soft - Via Rosellini, 12 - 20124 MILANO

COMPRO   VENDO   CAMBIO

VIC 20   C64   PERIF.   SOFTWARE

Nome .... Cognome ....

Via .... C.A.P ....

Città .... Tel. ....

SUPERVIC è bello, però... (ovvero suggerimenti, idee, critiche, richieste e tutto ciò che vi passa per la testa).

Nome .... Cognome ....

Via .... C.A.P ....

Città .... Tel. ....

COMMODORE

GIOCARE È UN AFFARE

I giochi del VIC 20...

2 ne paghi 4 ne prendi.*



(...li scegli tu. 4 games al prezzo di 2)

* *Validità fino al 30 giugno 1984*

Divertimento doppio per il tuo VIC 20! Oggi paghi 2 games e ne prendi 4, scegliendoli nella vasta gamma di giochi che trovi dal tuo rivenditore Commodore.

Ma affrettati, perché i favolosi giochi del VIC stanno andando a ruba... e fatti furbo: porta con te un amico e dividi con lui il costo delle cartucce; così ne porti via un mucchio, tutte originali, già pronte per giocare... e durano un secolo!

Con VIC 20 giocare è un affare.

Per altre informazioni, telefonaci subito: 02-618321.

Commodore Italiana S.p.A.

commodore COMPUTER

ETHOS